UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70210 ALDO HENRIQUE XAVIER 19HEL108 - 1 apos 19/09, 50% _____ / / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

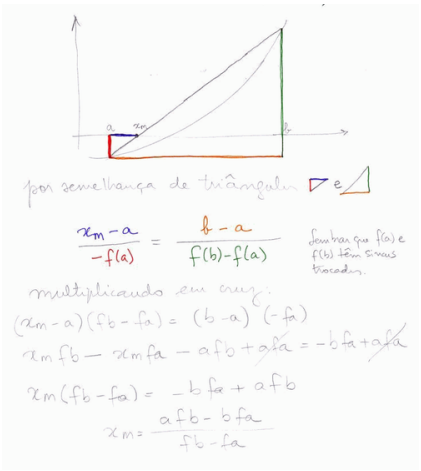
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

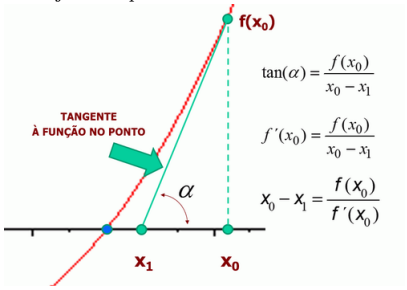
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
      {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[24]{5263}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 5*sin(x)+7*cos(x)-.637974 = 0$ no intervalo entre 8 e 9 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((3 * x)/10) * e^{((7*x)/10)} - 394.455840 = 0$ no intervalo entre 7 e 8 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7*x^6+3*x^2-1977203.585367 = 0$ no intervalo entre 8 e 9 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70227 ALVARO JOSE BARANOSKI 19HEL108 - 2 apos 19/09, 50%
/ _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

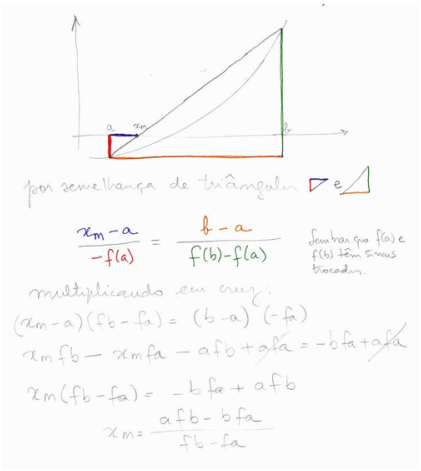
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

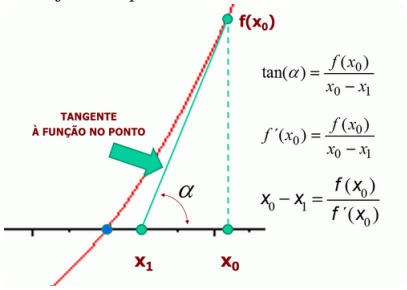
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
      {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[21]{9711}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 2 * sin(x) + 7 * cos(x) - 5.744596 = 0$ no intervalo entre 5 e 6 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((5 * x)/10) * e^{((4*x)/10)} - 3.677982 = 0$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 3 * x^2 - 1059499.454023 = 0$ no intervalo entre 7 e 8 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70234 ANAJULIA DE COUTO FLORIANO 19HEL108 - 3 apos 19/09, 50% _____ / _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

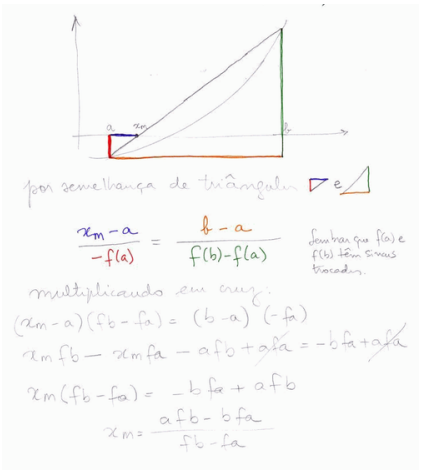
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

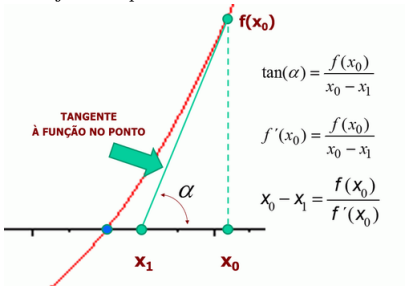
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
      {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[19]{6942}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 7 * sin(x) + 6 * cos(x) - 8.904511 = 0$ no intervalo entre 0 e 1 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((4 * x)/10) * e^{((6*x)/10)} - 26.783517 = 0$ no intervalo entre 4 e 5 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 4 * x^2 - 692257.338368 = 0$ no intervalo entre 6 e 7 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70827 ANDRE LUIS O. ANDRADE 19HEL108 - 4 apos 19/09, 50% / _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

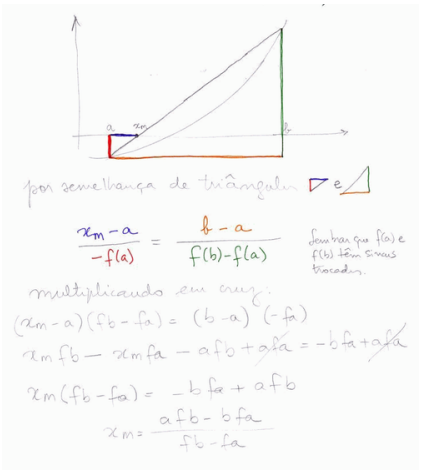
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

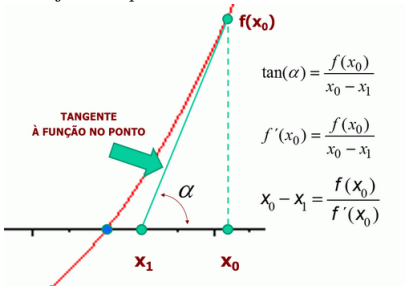
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
       {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[17]{6396}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 5*sin(x)+3*cos(x)-.588930 = 0$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((8 * x)/10) * e^{((7*x)/10)} - 1.351880 = 0$ no intervalo entre 0 e 1 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7*x^6+3*x^2-1977203.585367 = 0$ no intervalo entre 8 e 9 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70834 ANDRE LUIZ
DOMINGOS 19HEL108 - 5 apos 19/09, 50%
/ _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

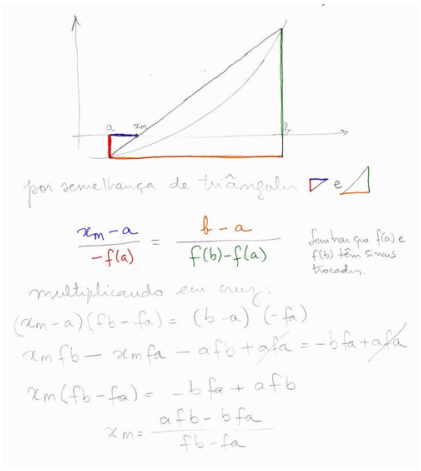
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

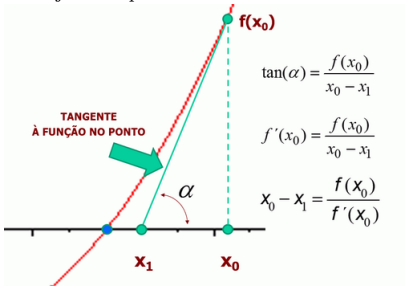
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
      {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[24]{5661}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 4*sin(x)+6*cos(x)+.418123 = 0$ no intervalo entre 8 e 9 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((5 * x)/10) * e^{((8*x)/10)} - 10.405810 = 0$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 5 * x^3 - 4.395008 = 0$ no intervalo entre 0 e 1 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70241 BIANCA LIMA DE OLIVEIRA 19HEL108 - 6 apos 19/09, 50%
/ _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

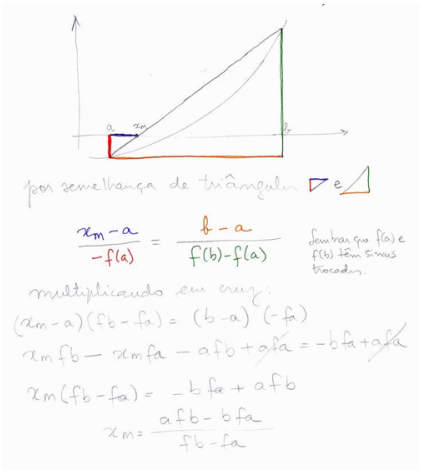
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

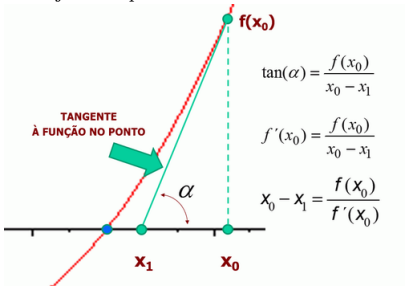
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
      {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[18]{9569}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 5 * sin(x) + 8 * cos(x) + 7.787223 = 0$ no intervalo entre 4 e 5 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((4 * x)/10) * e^{((5*x)/10)} - 62.803277 = 0$ no intervalo entre 6 e 7 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 5 * x^4 + 3 * x^2 - 25105.248000 = 0$ no intervalo entre 8 e 9 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70258 BRUNO MACHADO CARNEIRO 19HEL108 - 7 apos 19/09, 50%
/ _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

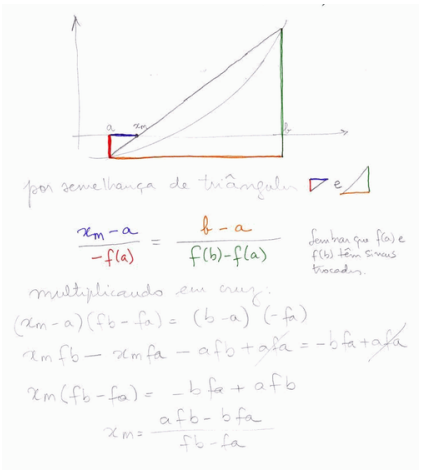
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
   -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

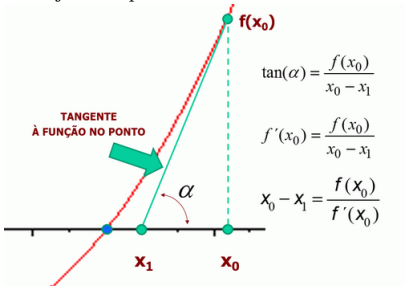
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
   -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
      {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[27]{5208}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 3*sin(x)+4*cos(x)-.071485 = 0$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((8 * x)/10) * e^{((6*x)/10)} - 672.485253 = 0$ no intervalo entre 7 e 8 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 5 * x^4 + 3 * x^2 - 490.590500 = 0$ no intervalo entre 3 e 4 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70265 CAIO HEITOR BARZ MOLINARI 19HEL108 - 8 apos 19/09, 50%
/ _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

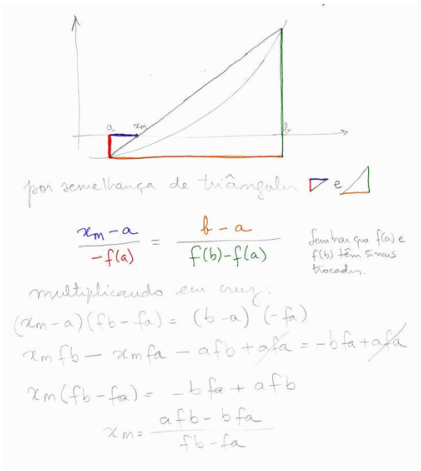
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
   -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

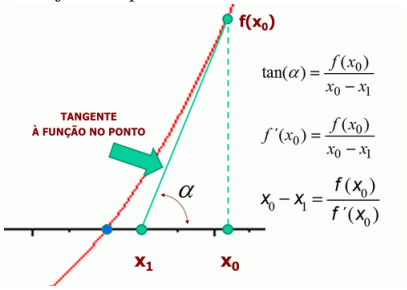
Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
   -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
       {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton

Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[25]{6939}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 4 * sin(x) + 6 * cos(x) - 5.170961 = 0$ no intervalo entre 6 e 7 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((6 * x)/10) * e^{((3*x)/10)} - 1.853287 = 0$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 4 * x^3 - 2834514.869952 = 0$ no intervalo entre 8 e 9 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 71156 CAIO MAGANHINI LOPES 19HEL108 - 9 apos 19/09, 50% _____ / / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

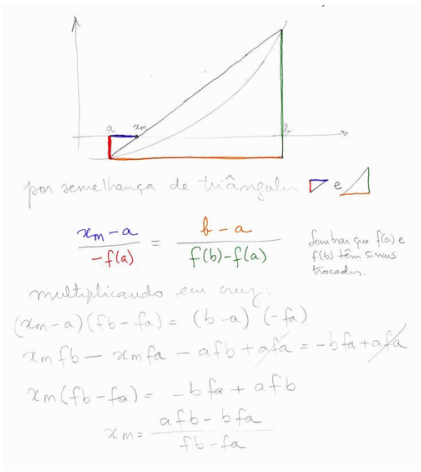
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
   -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

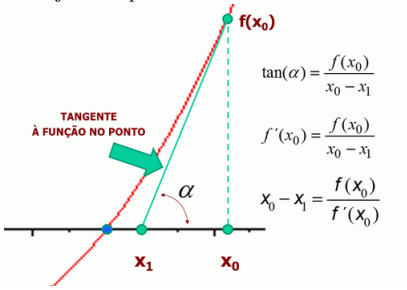
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
   -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
       {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[14]{7078}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 5 * sin(x) + 6 * cos(x) + 5.073948 = 0$ no intervalo entre 4 e 5 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((5 * x)/10) * e^{((6*x)/10)} - 13.074036 = 0$ no intervalo entre 3 e 4 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^4 + 3 * x^2 - 13412.995200 = 0$ no intervalo entre 6 e 7 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70272 DANIEL SANTOS DA SILVA 19HEL108 - 10 apos 19/09, 50% _____ / / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

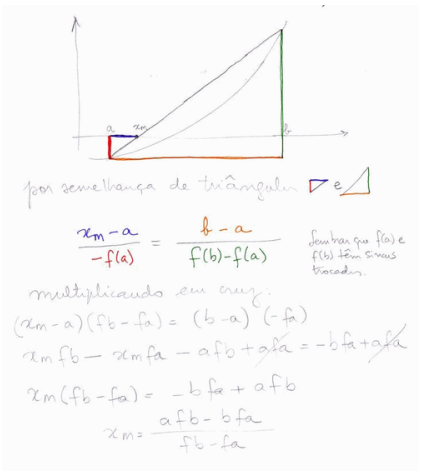
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
   -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

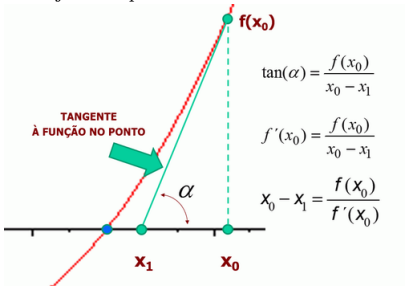
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
   -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
      {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[9]{6624}$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 4*sin(x)+5*cos(x)+.557198 = 0$ no intervalo entre 5 e 6 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((5 * x)/10) * e^{((7*x)/10)} - 1.865119 = 0$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 4 * x^2 - 58207.359375 = 0$ no intervalo entre 4 e 5 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70289 ELEN DE JESUS SANTANA 19HEL108 - 11 apos 19/09, 50%
/ _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

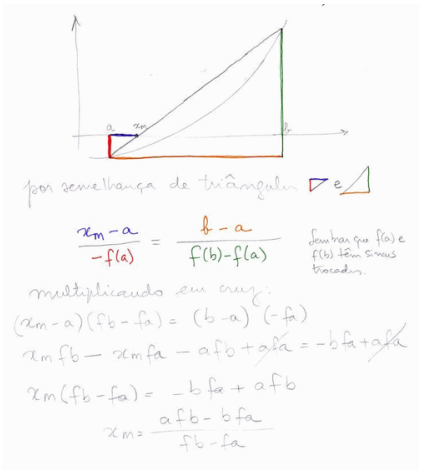
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

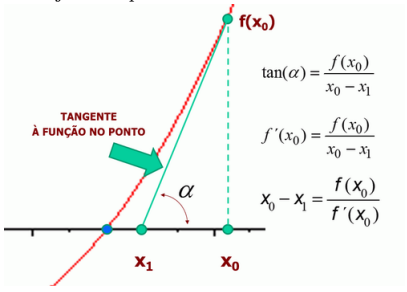
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
      {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[13]{7889}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 8 * sin(x) + 4 * cos(x) + 5.193571 = 0$ no intervalo entre 5 e 6 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((7 * x)/10) * e^{((8*x)/10)} - .857830 = 0$ no intervalo entre 0 e 1 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 5 * x^4 - 220804.104192 = 0$ no intervalo entre 5 e 6 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70841 FELIPE NAVES MENDONCA 19HEL108 - 12 apos 19/09, 50% / _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

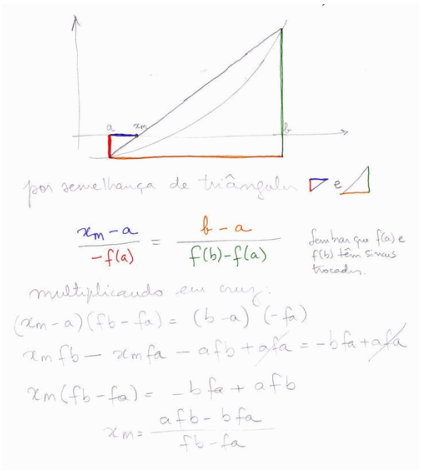
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

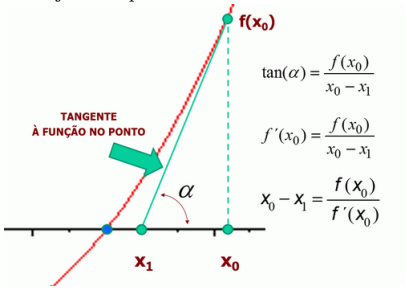
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
      {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[22]{6655}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 8 * sin(x) + 3 * cos(x) - 5.391137 = 0$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((8 * x)/10) * e^{((4*x)/10)} - 10.597646 = 0$ no intervalo entre 3 e 4 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 4 * x^2 - 75542.867303 = 0$ no intervalo entre 4 e 5 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 71251 GIOVANA ASSIS
JIENTARA 19HEL108 - 13 apos 19/09, 50%
/ _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

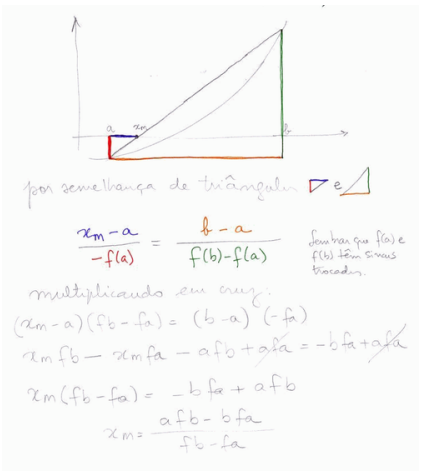
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
   -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

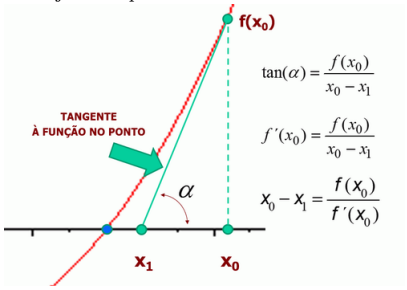
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
   -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
       {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[7]{9710}$ no intervalo entre 3 e 4 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 7 * sin(x) + 6 * cos(x) - 2.866460 = 0$ no intervalo entre 8 e 9 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((4 * x)/10) * e^{((5*x)/10)} - 4.945213 = 0$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 6 * x^5 + 4 * x^2 - 11152.687500 = 0$ no intervalo entre 4 e 5 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70296 GIOVANNI MOTA
BRIGATI 19HEL108 - 14 apos 19/09, 50%
/ _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

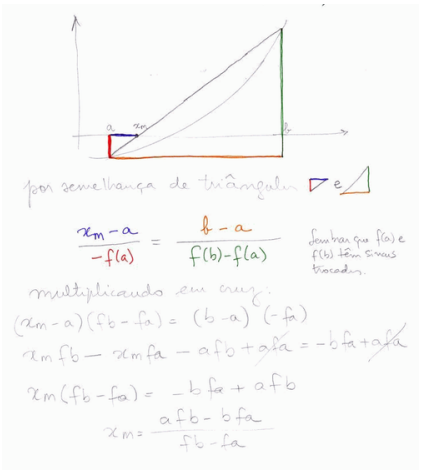
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
   -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

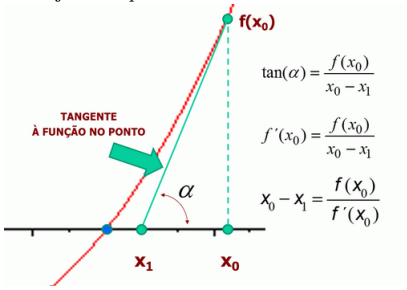
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
   -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
      {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[6]{6851}$ no intervalo entre 4 e 5 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 4 * sin(x) + 3 * cos(x) - 2.314557 = 0$ no intervalo entre 8 e 9 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((3 * x)/10) * e^{((4*x)/10)} - 4.257960 = 0$ no intervalo entre 3 e 4 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^5 + 4 * x^3 - 11884.871680 = 0$ no intervalo entre 4 e 5 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70308 GUILHERME FONTES DUARTE 19HEL108 - 15 apos 19/09, 50% _____ / _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

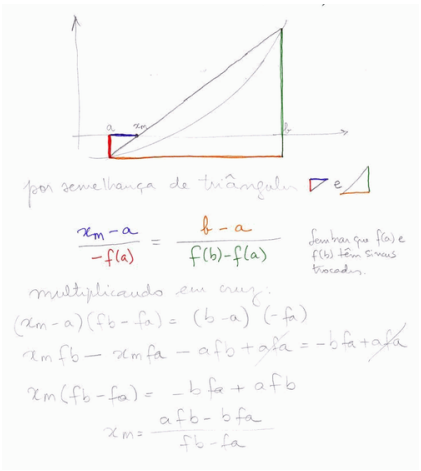
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

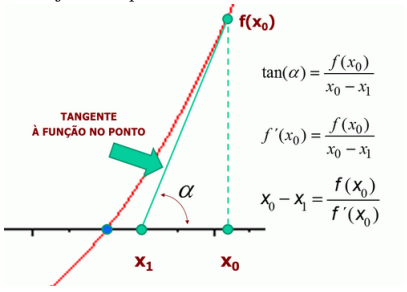
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
      {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[24]{9277}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 6 * sin(x) + 5 * cos(x) - 2.394132 = 0$ no intervalo entre 5 e 6 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((7 * x)/10) * e^{((2*x)/10)} - 13.440565 = 0$ no intervalo entre 5 e 6 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 5 * x^4 - 220804.104192 = 0$ no intervalo entre 5 e 6 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70315 HENRIQUE DOMICIANO OSINSKI 19HEL108 - 16 apos 19/09, 50% _____ / _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

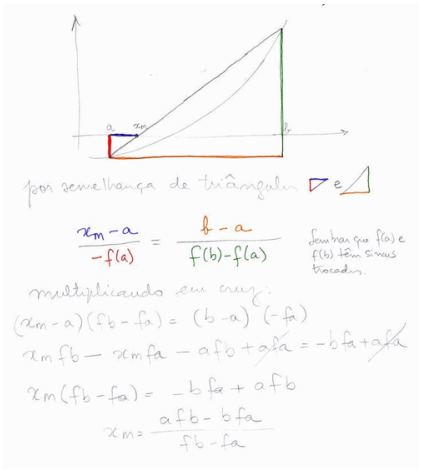
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

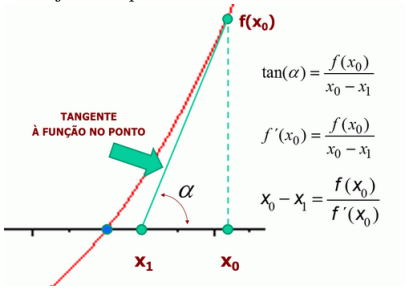
Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
       {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton

Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[7]{8527}$ no intervalo entre 3 e 4 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 6 * sin(x) + 2 * cos(x) - 6.252633 = 0$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((7 * x)/10) * e^{((8*x)/10)} - 32.370500 = 0$ no intervalo entre 3 e 4 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 5 * x^4 + 3 * x^2 - 44546.910500 = 0$ no intervalo entre 9 e 10 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70322 HENRIQUE MARTINS DOMANSKI 19HEL108 - 17 apos 19/09, 50%
/ _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

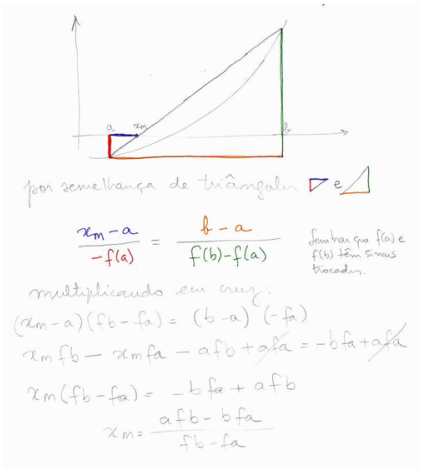
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

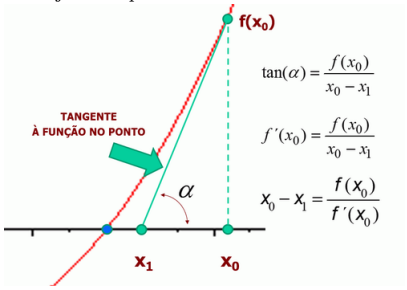
Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
      {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton

Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[18]{7145}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 4 * sin(x) + 5 * cos(x) - 2.168753 = 0$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((4 * x)/10) * e^{((3*x)/10)} - 8.524501 = 0$ no intervalo entre 4 e 5 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 5 * x^4 + 3 * x^2 - 9052.062500 = 0$ no intervalo entre 6 e 7 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70339 HERICK DENKE
19HEL108 - 18 apos 19/09, 50% _____ / / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

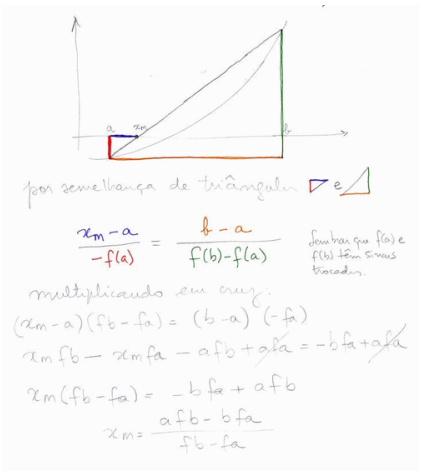
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

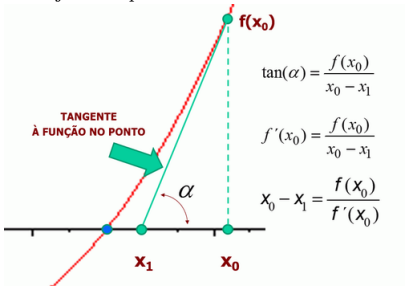
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
       {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[26]{7624}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 5 * sin(x) + 2 * cos(x) - 4.523649 = 0$ no intervalo entre 6 e 7 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((7 * x)/10) * e^{((3*x)/10)} - 4.540079 = 0$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 6 * x^4 + 3 * x^2 - 129.918600 = 0$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70346 IURI PAVANI DE SOUZA 19HEL108 - 19 apos 19/09, 50% _____ / / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção

Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

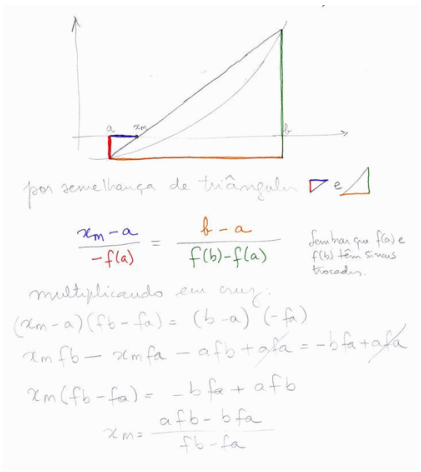
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
   -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

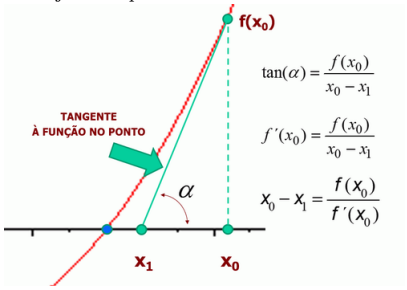
Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
   -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
       {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton

Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[23]{9997}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 5 * sin(x) + 2 * cos(x) + 2.370100 = 0$ no intervalo entre 9 e 10 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((3 * x)/10) * e^{((5*x)/10)} - 178.768801 = 0$ no intervalo entre 8 e 9 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 6 * x^4 + 3 * x^2 - 28.929600 = 0$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70858 JHONATAN K. RIBEIRO 19HEL108 - 20 apos 19/09, 50%
/ _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

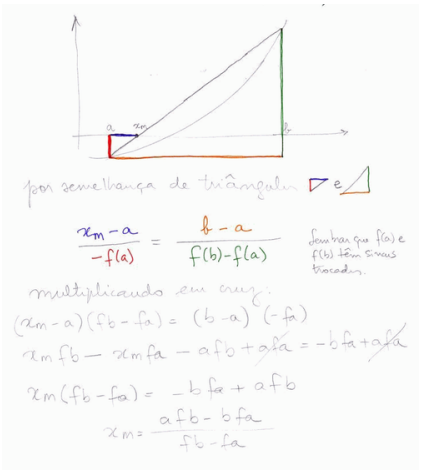
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

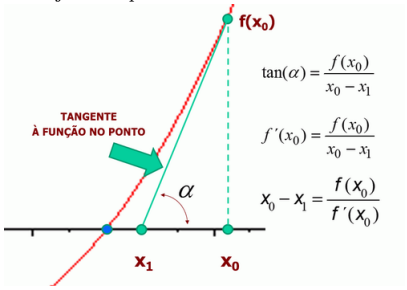
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
      {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[19]{6970}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 8*sin(x)+7*cos(x)-10.623579 = 0$ no intervalo entre 7 e 8 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((8 * x)/10) * e^{((6*x)/10)} - 581.147557 = 0$ no intervalo entre 7 e 8 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 4 * x^3 - 6.636087 = 0$ no intervalo entre 0 e 1 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70353 JODERSON GOMES DA SILVA 19HEL108 - 21 apos 19/09, 50%
/ _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

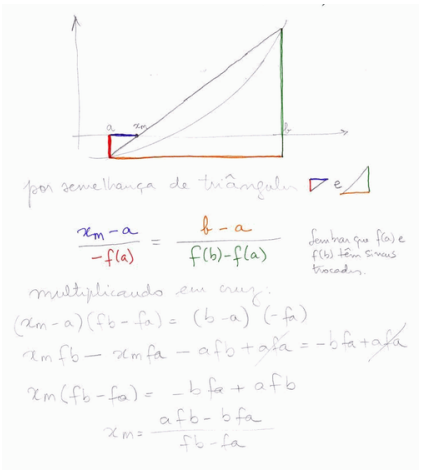
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

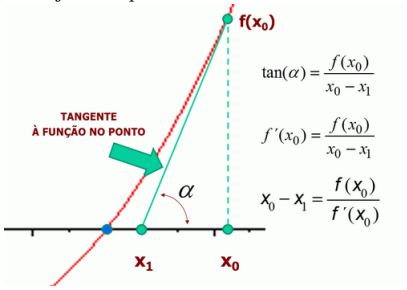
Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
       {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton

Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[20]{8213}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 3 * sin(x) + 8 * cos(x) - 8.436541 = 0$ no intervalo entre 0 e 1 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((5 * x)/10) * e^{((2*x)/10)} - .762749 = 0$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 6 * x^5 + 3 * x^2 - 3666.850560 = 0$ no intervalo entre 3 e 4 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70360 LEONARDO CORNELIO KUIPERS 19HEL108 - 22 apos 19/09, 50% _____ / _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

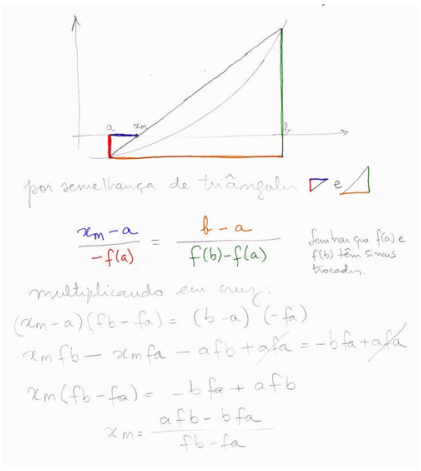
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
   -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

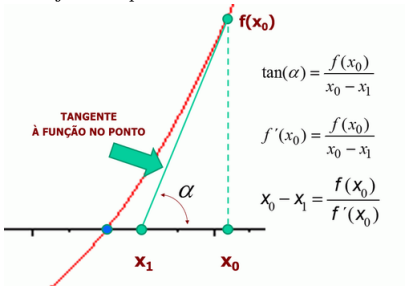
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
   -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
       {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[17]{8350}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 7 * sin(x) + 4 * cos(x) + 4.514746 = 0$ no intervalo entre 9 e 10 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((7 * x)/10) * e^{((8*x)/10)} - 11084.903940 = 0$ no intervalo entre 9 e 10 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 6 * x^5 + 4 * x^2 - 2.968420 = 0$ no intervalo entre 0 e 1 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70377 LUCAS ANTONIO THEODORO SANTOS 19HEL108 - 23 apos 19/09, 50% _____ / _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

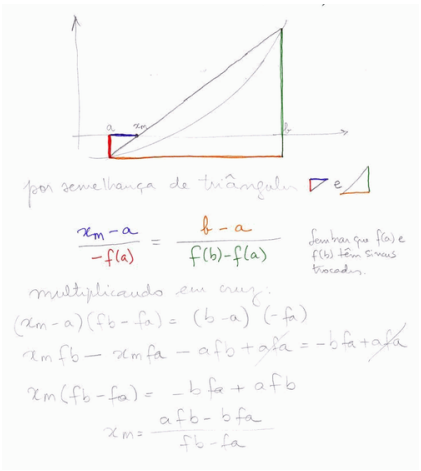
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

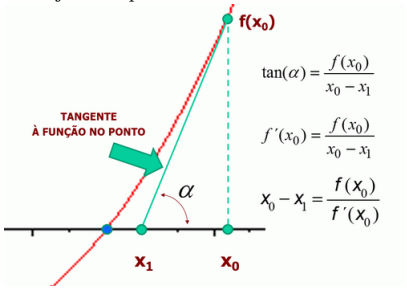
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
      {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[21]{9529}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 2 * sin(x) + 6 * cos(x) - 6.192208 = 0$ no intervalo entre 6 e 7 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((8 * x)/10) * e^{((2*x)/10)} - 49.282357 = 0$ no intervalo entre 9 e 10 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 4 * x^3 - 3038017.419063 = 0$ no intervalo entre 8 e 9 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70384 LUCAS BOHN
DUTRA 19HEL108 - 24 apos 19/09, 50%
/ _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

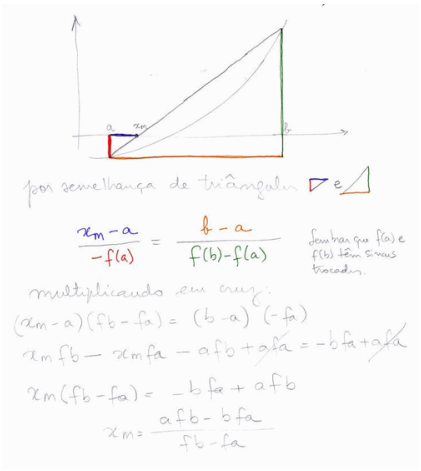
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

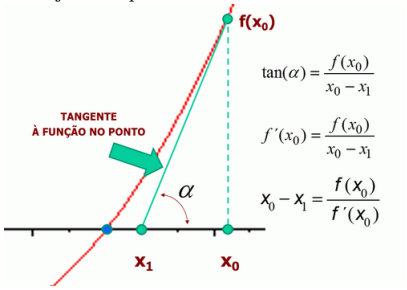
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
       {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[15]{8620}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 4 * sin(x) + 8 * cos(x) + 8.277982 = 0$ no intervalo entre 9 e 10 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((3 * x)/10) * e^{((4*x)/10)} - .386999 = 0$ no intervalo entre 0 e 1 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 4 * x^3 - 13039.359375 = 0$ no intervalo entre 3 e 4 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70391 LUCAS CORDEIRO NEVES 19HEL108 - 25 apos 19/09, 50% _____ / / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

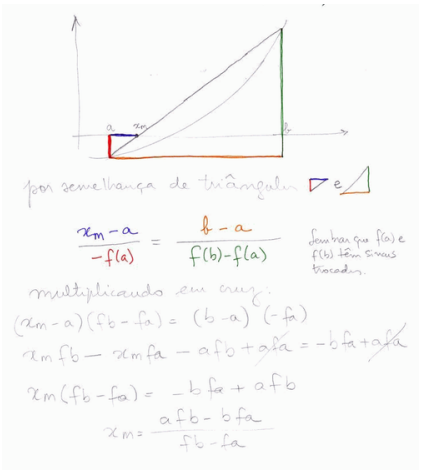
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

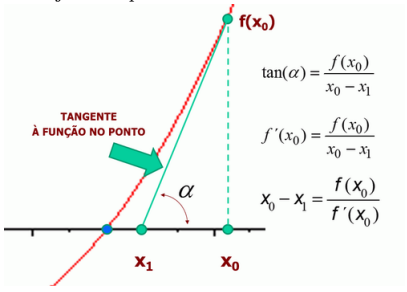
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
      {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[10]{9518}$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 2 * sin(x) + 8 * cos(x) + 8.215330 = 0$ no intervalo entre 3 e 4 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((6 * x)/10) * e^{((3*x)/10)} - .270599 = 0$ no intervalo entre 0 e 1 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 5 * x^4 + 3 * x^2 - 42743.808000 = 0$ no intervalo entre 9 e 10 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70403 LUCAS FAGANELLO SILVA 19HEL108 - 26 apos 19/09, 50% _____ / / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

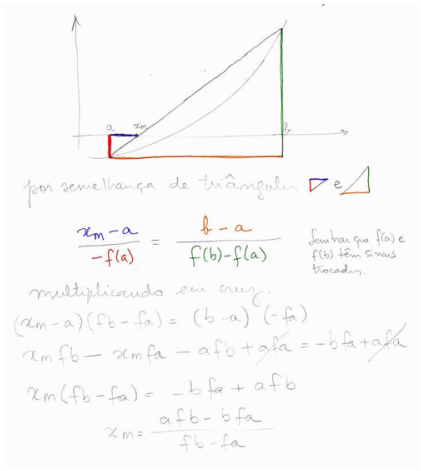
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

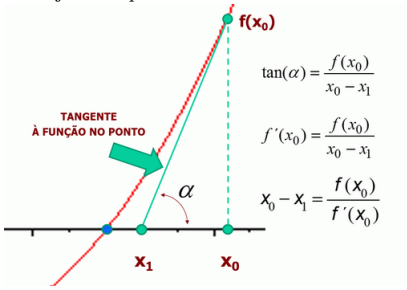
Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
       {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton

Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[11]{6938}$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 6 * sin(x) + 4 * cos(x) + 7.100192 = 0$ no intervalo entre 4 e 5 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((4 * x)/10) * e^{((5*x)/10)} - 197.916143 = 0$ no intervalo entre 8 e 9 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 4 * x^2 - 5146004.234375 = 0$ no intervalo entre 9 e 10 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70410 LUIZ AUGUSTO SHOITI TSURUKAWA 19HEL108 - 27 apos 19/09, 50% _____ / _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

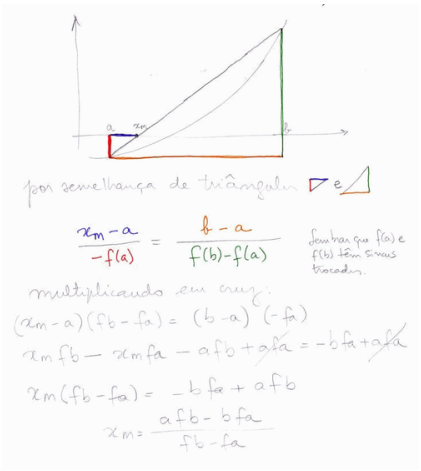
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
   -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

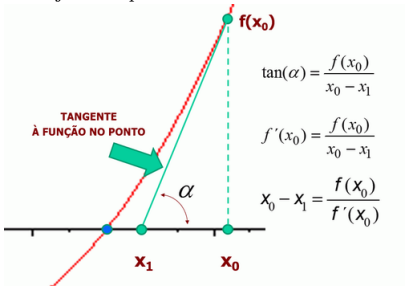
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
   -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
       {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[21]{9204}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 2 * sin(x) + 3 * cos(x) + 3.575316 = 0$ no intervalo entre 3 e 4 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((2 * x)/10) * e^{((6*x)/10)} - 371.152624 = 0$ no intervalo entre 8 e 9 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 6 * x^4 + 3 * x^2 - 6431.070600 = 0$ no intervalo entre 5 e 6 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70427 MARCOS A. DA COSTA 19HEL108 - 28 apos 19/09, 50% _____ / / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

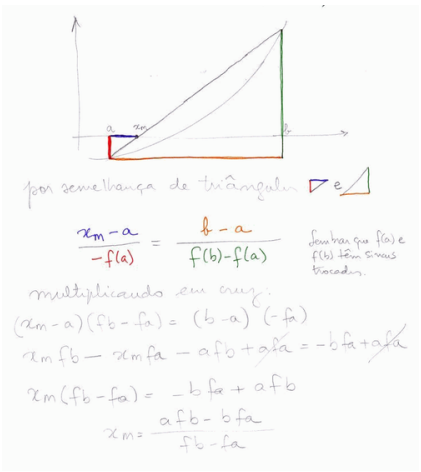
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
   -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

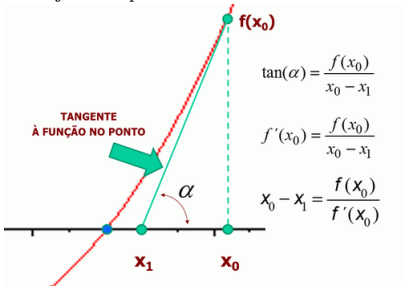
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
   -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
      {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[21]{5457}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 8 * sin(x) + 4 * cos(x) - 4.759637 = 0$ no intervalo entre 8 e 9 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((7 * x)/10) * e^{((4*x)/10)} - 197.679707 = 0$ no intervalo entre 8 e 9 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^5 + 3 * x^2 - 75284.807680 = 0$ no intervalo entre 6 e 7 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70434 MARIANA YAEDA ARTUSO 19HEL108 - 29 apos 19/09, 50%
/ _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

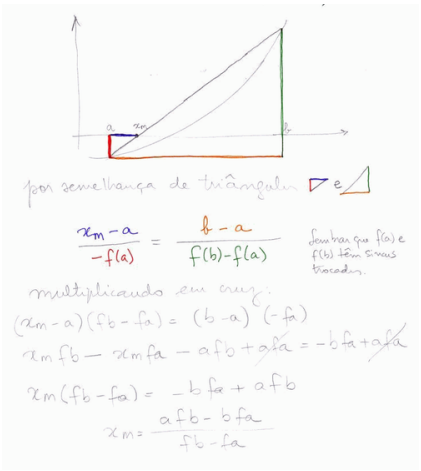
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

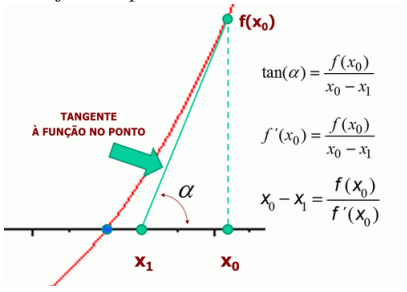
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
      {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[20]{5457}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 3 * sin(x) + 5 * cos(x) + 5.790412 = 0$ no intervalo entre 3 e 4 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((7 * x)/10) * e^{((4*x)/10)} - 2.843907 = 0$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 5 * x^4 - 290.573568 = 0$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 71220 MATHEUS CARDOSO CASTILHO DIAS 19HEL108 - 30 apos 19/09, 50% _____ / _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção

Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

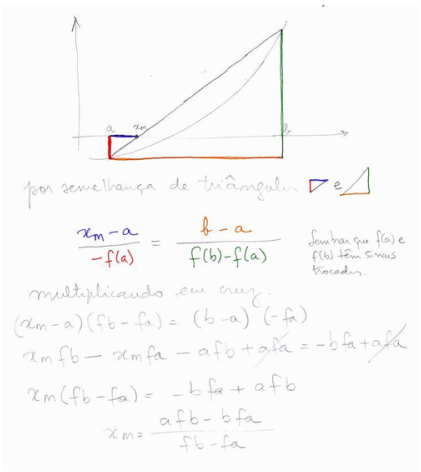
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
   -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

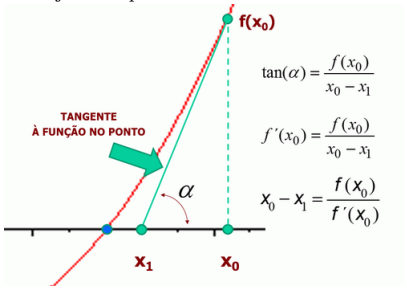
Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
   -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
       {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton

Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[24]{7062}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 3 * sin(x) + 4 * cos(x) - 4.836421 = 0$ no intervalo entre 0 e 1 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((6 * x)/10) * e^{((7*x)/10)} - 201.750603 = 0$ no intervalo entre 5 e 6 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 6 * x^5 + 4 * x^2 - 739.922560 = 0$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70441 MATHEUS DA SILVA QUADROS 19HEL108 - 31 apos 19/09, 50%
/ _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

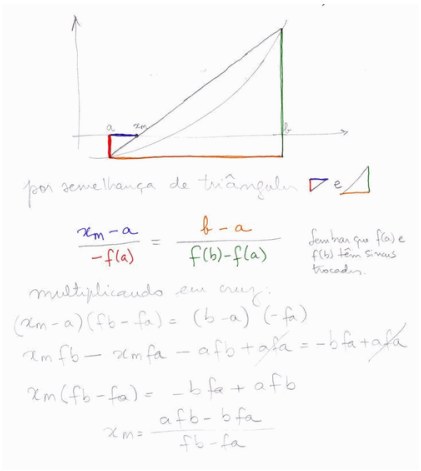
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
   -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

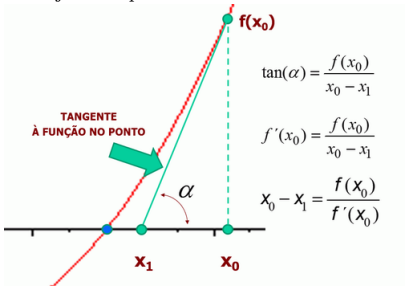
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
   -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
      {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[17]{7003}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 4*sin(x)+6*cos(x)+.722719 = 0$ no intervalo entre 5 e 6 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((7 * x)/10) * e^{((6*x)/10)} - 508.504113 = 0$ no intervalo entre 7 e 8 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7*x^6+3*x^2-5831086.304503 = 0$ no intervalo entre 9 e 10 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70458 MICCAELA BOGASZ SANTIAGO 19HEL108 - 32 apos 19/09, 50%
/ _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

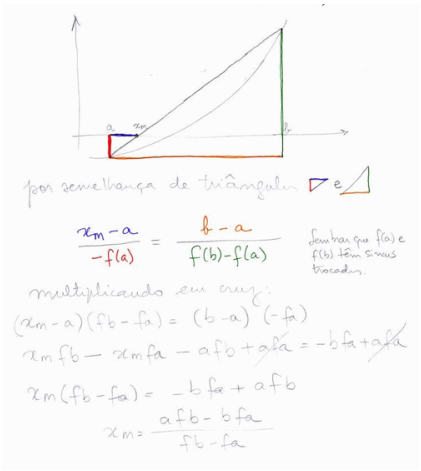
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

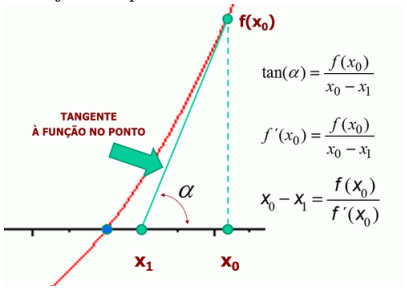
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
       {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[18]{8894}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 7 * sin(x) + 5 * cos(x) - 2.020255 = 0$ no intervalo entre 5 e 6 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((5 * x)/10) * e^{((8*x)/10)} - 35.701248 = 0$ no intervalo entre 3 e 4 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^5 + 4 * x^3 - 179242.680320 = 0$ no intervalo entre 7 e 8 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 71194 NATALIA GONCALVES P CHOMA 19HEL108 - 33 apos 19/09, 50% _____ / _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

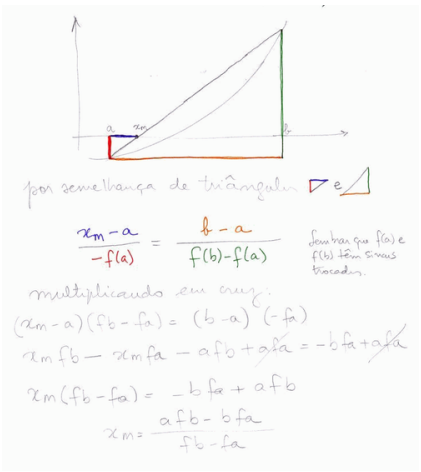
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

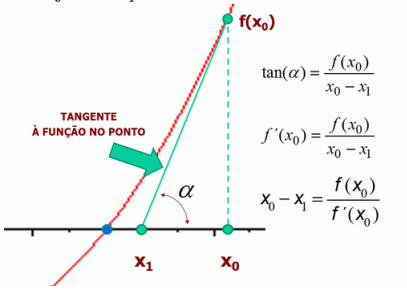
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
       {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[8]{5975}$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 2 * sin(x) + 7 * cos(x) - 1.081418 = 0$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((7 * x)/10) * e^{((4*x)/10)} - 4.039958 = 0$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 3 * x^2 - 4829353.547392 = 0$ no intervalo entre 9 e 10 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

108-71194 - 19/09

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70465 PEDRO AFONSO CARDOSO SZPAK 19HEL108 - 34 apos 19/09, 50% _____ / _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

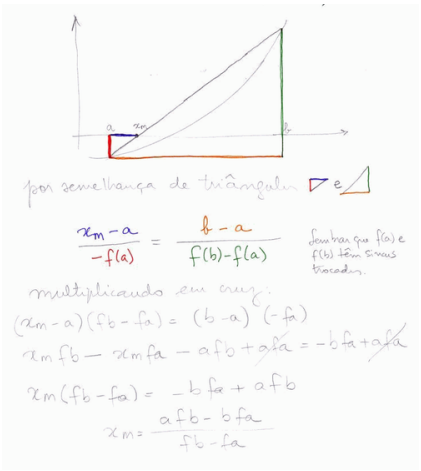
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

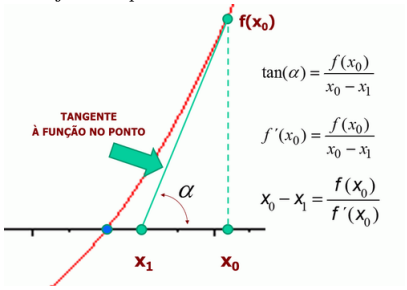
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
      {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[12]{5093}$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 2 * sin(x) + 8 * cos(x) - 8.224943 = 0$ no intervalo entre 6 e 7 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((4 * x)/10) * e^{((8*x)/10)} - 179.191911 = 0$ no intervalo entre 5 e 6 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 4 * x^2 - 2741.103423 = 0$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 71163 PEDRO HENRIQUE L ARAUJO 19HEL108 - 35 apos 19/09, 50%
/ _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

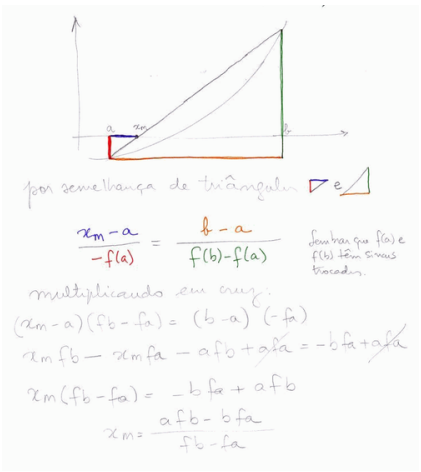
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

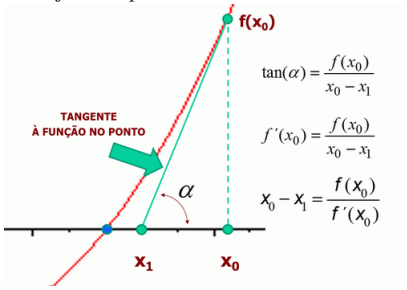
Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
       {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton

Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[15]{5591}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 4 * sin(x) + 8 * cos(x) - 1.049683 = 0$ no intervalo entre 8 e 9 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((4 * x)/10) * e^{((8*x)/10)} - 7.389056 = 0$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 4 * x^2 - 40.547663 = 0$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70472 RODOLPHO DE SOUZA DAVID 19HEL108 - 36 apos 19/09, 50% / _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

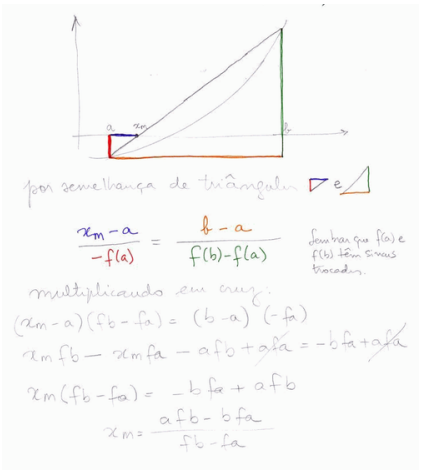
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
   -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

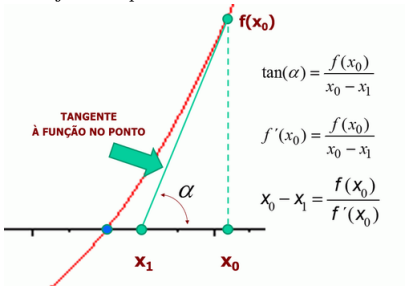
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
   -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
      {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[9]{9669}$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 8 * sin(x) + 3 * cos(x) - 7.546785 = 0$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((7 * x)/10) * e^{((6*x)/10)} - 378.950687 = 0$ no intervalo entre 7 e 8 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 5 * x^4 - 45958.941843 = 0$ no intervalo entre 4 e 5 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70489 TAYANE CORDEIRO FARIA 19HEL108 - 37 apos 19/09, 50% _____ / / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

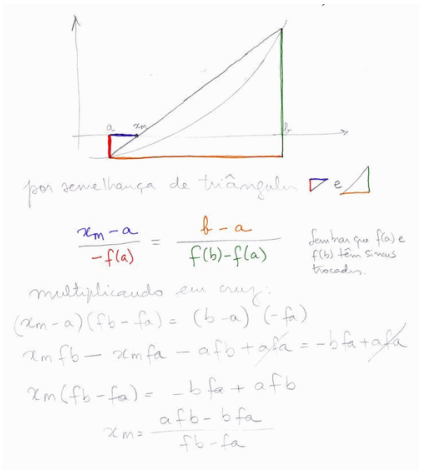
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

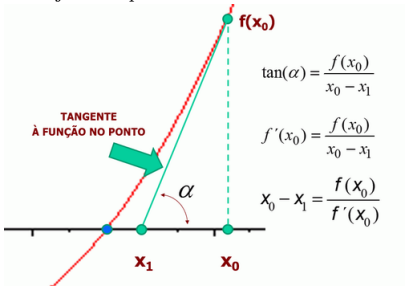
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
      {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[14]{7527}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 2 * sin(x) + 4 * cos(x) - 3.568749 = 0$ no intervalo entre 6 e 7 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((8 * x)/10) * e^{((4*x)/10)} - 2.684474 = 0$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 4 * x^2 - 975405.846528 = 0$ no intervalo entre 7 e 8 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70496 THIAGO ENRIQUE B. FIORESE 19HEL108 - 38 apos 19/09, 50%
/ _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

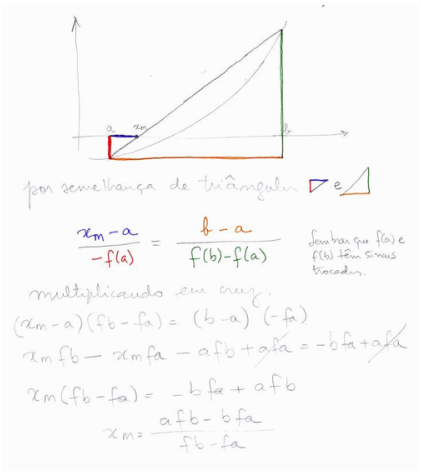
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

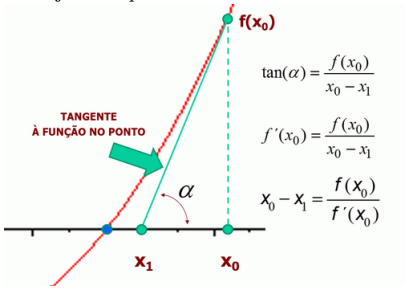
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
      {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[22]{9078}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 6 * sin(x) + 4 * cos(x) + 6.242156 = 0$ no intervalo entre 3 e 4 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((5 * x)/10) * e^{((8*x)/10)} - 150.820948 = 0$ no intervalo entre 5 e 6 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 5 * x^3 - 29.541888 = 0$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70515 VALTER
WERGENSKI 19HEL108 - 39 apos 19/09, 50%
/ _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

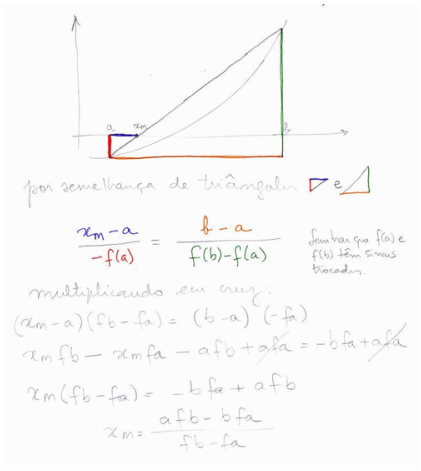
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

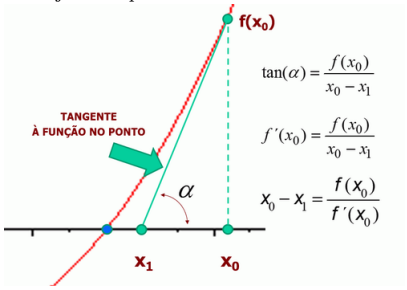
Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
       {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton

Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[24]{5866}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 8 * sin(x) + 4 * cos(x) + 8.405858 = 0$ no intervalo entre 3 e 4 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((4 * x)/10) * e^{((3*x)/10)} - .612026 = 0$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 5 * x^2 - 975457.686528 = 0$ no intervalo entre 7 e 8 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70522 VICTOR EMANUEL VENTURI 19HEL108 - 40 apos 19/09, 50%
/ _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

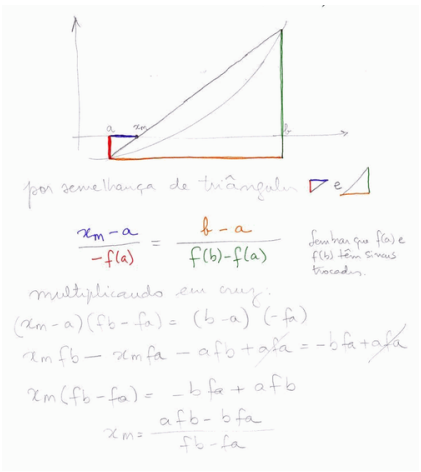
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

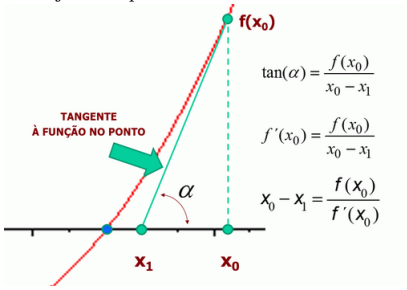
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
       {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[22]{5142}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 8 * sin(x) + 3 * cos(x) + 1.141412 = 0$ no intervalo entre 9 e 10 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((4 * x)/10) * e^{((6*x)/10)} - .851309 = 0$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 5 * x^2 - .200448 = 0$ no intervalo entre 0 e 1 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70539 VINICIUS
GUILHERME F. SILVA 19HEL108 - 41 apos 19/09, 50% _____ / _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

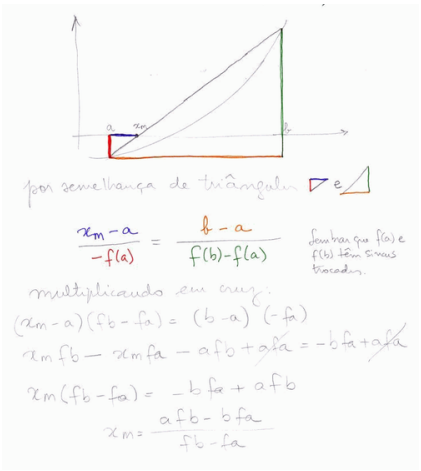
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

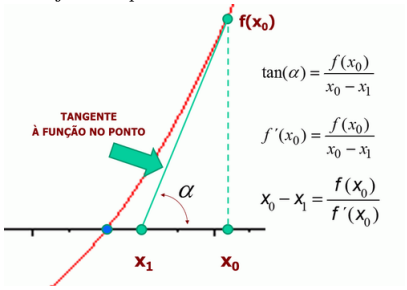
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
      {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[24]{7684}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 8 * sin(x) + 3 * cos(x) - 4.581861 = 0$ no intervalo entre 8 e 9 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((5 * x)/10) * e^{((6*x)/10)} - .646430 = 0$ no intervalo entre 0 e 1 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 5 * x^2 - 24707.256327 = 0$ no intervalo entre 3 e 4 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:33:06.9
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 71244 VINICIUS LUIS ALFIERI 19HEL108 - 42 apos 19/09, 50%
/ _____ / _____

Cálculo Numérico - solução de equações

Suponha que precisamos usar o computador para resolver equações. Já paramos para entender o que é resolver uma equação ? Conhece-se a forma geral de uma equação $y = f(x)$. Resolver esta equação vem a ser descobrir quais os valores de x que fazem $f(x)$ ser igual a zero. Se olharmos para o gráfico, veremos que busca-se o valor da abcissa x que determina o cruzamento da curva de $f(x)$ no eixo dos x.

Por exemplo, suponha-se precisar descobrir a raiz 11 do número 1739 ? Ou seja, qual o número que multiplicado por ele mesmo 11 vezes dá como resultado 1739 ?

Dizendo isso na forma matemática, pode-se escrever $x^{11} = 1739$, ou melhor ainda $x^{11} - 1739 = 0$. Com isso, chegamos ao ponto.

Método da Bissecção Se olharmos o gráfico de uma função, ver-se-á que no entorno da raiz, de um lado a função é positiva, e do outro lado é negativa ou vice versa. Este método começa pela seleção de 2 pontos x_a e x_b sendo que a função $y = f(x)$ têm sinais opostos nesses dois pontos. Graças a essa constatação, pode-se afirmar que a raiz procurada está entre x_a e x_b. Dizendo de outra maneira. O valor de x procurado, está entre a e b. O Método da bissecção vai dividindo o intervalo a, b pela metade, questionando a seguir em qual das metades o x está. Descoberto isso, despreza-se a metade que não contém x. Levando este processo ao limite, descobre-se qual o valor de x. Em resumo, eis as etapas do método:

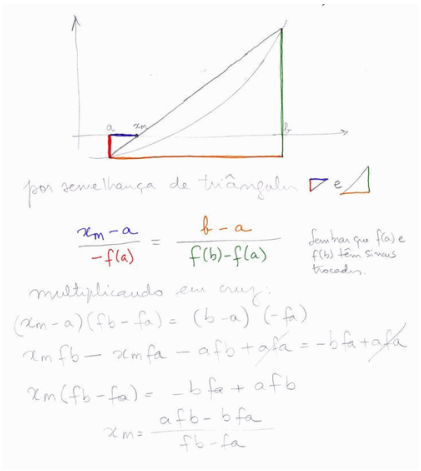
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

Seja um exemplo, calcular $f = e^x - sin(x) - 2 = 0$ nos limites $a = 1$ e $b = 1.25$ com $\epsilon = 0.01$. No primeira etapa tem-se

```
Limite Superior: 1.25
Limite Inferior: 1
Tolerancia: 0.01
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.1250  1.0000  1.1250  0.1779  0.5414 -0.1232  0.1250
  1.0625  1.0000  1.0625  0.0200  0.1779 -0.1232  0.0625
  1.0625  1.0312  1.0312 -0.0534  0.0200 -0.1232  0.0312
  1.0625  1.0469  1.0469 -0.0171  0.0200 -0.0534  0.0156
  1.0547  1.0469  1.0547  0.0013  0.0200 -0.0171  0.0078
Raiz:  1.05078125
Erro:  -0.0078125
Eis o programa Python que resolve este caso
import numpy as np
def bissec():
  a=float(input("Limite Superior: "))
  b=float(input("Limite Inferior: "))
  tol=float(input("Tolerancia: "))
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while np.abs(b-a)>tol:
    xm=(a+b)/2
    fxm=np.exp(xm)-np.sin(xm)-2
    fa=np.exp(a)-np.sin(a)-2
    fb=np.exp(b)-np.sin(b)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
   {:8.4f}{:8.4f}'.format(a,b,xm,fxm,fa,fb,np.abs(b-a)))
  xm=(a+b)/2
  print("Raiz: ",xm)
  print("Erro: ",b-a)

bissec()
```

Cordas ou Falsa Posição



O método começa com uma análise do gráfico da função nas proximidades da raiz. Por semelhança de triângulos, pode-se obter a formulação proposta. Eis os passos do algoritmo:

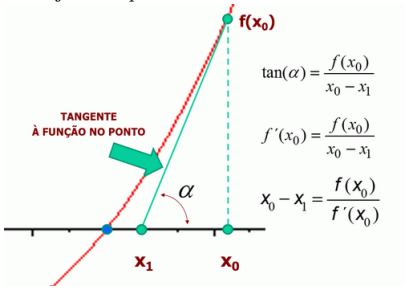
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
Limite inferior: 1
Limite superior: 1.25
Tolerância: 0.001
  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro
  1.0463  1.2500  1.0463 -0.0184 -0.1232  0.5414
  1.0530  1.2500  1.0530 -0.0026 -0.0184  0.5414
  1.0540  1.2500  1.0540 -0.0004 -0.0026  0.5414
Raiz:  1.0539728656764054
Erro:  -0.0003663936835978099

Eis o programa Python que resolve este caso
from numpy import *
def cordas():
  a=float(input("Limite inferior: "))
  b=float(input("Limite superior: "))
  tol=float(input("Tolerância: "))
  fxm=1
  print("  -----a  -----b  ----xm  ---fxm  ----fa  ----fb  --erro")
  while abs(fxm)>tol:
    fa=exp(a)-sin(a)-2
    fb=exp(b)-sin(b)-2
    xm=((a*fb)-(b*fa))/(fb-fa)
    fxm=exp(xm)-sin(xm)-2
    if fa*fxm>0:
      a=xm
    else:
      b=xm
    print('{:8.4f}{:8.4f}{:8.4f}{:8.4f}{:8.4f}
       {:8.4f}'.format(a,b,xm,fxm,fa,fb))
  print("Raiz: ",xm)
  print("Erro: ",fxm)
cordas()
```

Método de Newton Este método começa com a interpretação geométrica da derivada. Lembrando, Newton é um monstro da ciência. Entre outras coisas, ele criou o cálculo diferencial e integral, a teoria da luz, a gravitação, as leis do movimento, além de, no fim da vida ter – como funcionário público – dirigido a casa da moeda inglesa, pendurando na forca alguns falsários delinquentes. Voltando ao cálculo diferencial, uma das interpretações da derivada de uma função em um ponto é a tangente da curva da função naquele ponto. Veja-se o que se disse neste desenho



Eis as etapas do método:

1. Obter a derivada da função a resolver
2. Arbitrar um ponto inicial x_0 para a função
3. Arbitrar o erro ϵ
4. Calcular $f(x_0)$ e $df(x_0)$
5. Calcular o "novo" x, $x = x_0 - fx/dfx$
6. Se $|fx| > \epsilon$ retornar ao passo 4

Por exemplo, seja calcular as raízes da mesma equação acima que é $e^x - sen(x) - 2 = 0$. A derivada é $\frac{\partial}{\partial x} = e^x - cos(x)$. Obteve-se a seguinte tabela

```
   --------x   -----------------fx   -------dfx
   1.0000000 -0.12318915634885141    2.1779795
   1.0565612  0.00579321190120519    2.3845934
   1.0541318  0.00001105001756940    2.3754999
   1.0541271  0.00000000004045120    2.3754825
Raiz:  1.0541271241082415
Erro:  4.0451197946822504e-11
Eis o programa Python que resolve este caso
from numpy import *
def newton1():
  x=1
  tol=0.00001
  fx=exp(x)-sin(x)-2
  dfx=exp(x)-cos(x)
  print("   --------x   -----------------fx   -------dfx")
  print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  while abs(fx)>tol:
    x=x-fx/dfx
    fx=exp(x)-sin(x)-2
    dfx=exp(x)-cos(x)
    print("{:12.7f} {:20.17f} {:12.7f}".format(x,fx,dfx))
  print("Raiz: ",x)
  print("Erro: ",fx)
newton1()
```

☞ Para você fazer

1. Resolva pelos 3 métodos, para poder comparar, a seguinte equação $x = \sqrt[7]{9759}$ no intervalo entre 3 e 4 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 5 * sin(x) + 7 * cos(x) - 2.468473 = 0$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((6 * x)/10) * e^{((5*x)/10)} - .716076 = 0$ no intervalo entre 0 e 1 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 6 * x^4 + 3 * x^2 - 58.782600 = 0$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	