UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70865 ALEXANDRE S A DELDUQUE MACEDO 19HEA108 - 1 apos
18/09, 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:

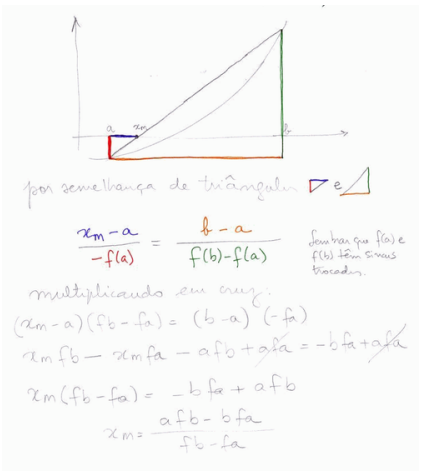
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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

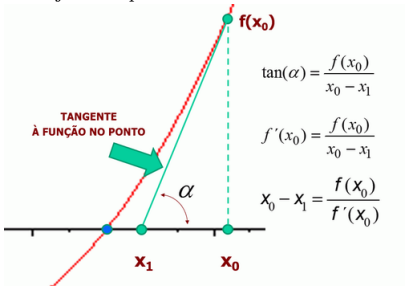
Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
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 $x = \sqrt[22]{7572}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 8 * sin(x) + 3 * cos(x) - 7.074693 = 0$ no intervalo entre 6 e 7 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((6 * x)/10) * e^{((8*x)/10)} - 2.206789 = 0$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 3 * x^2 - 808.179328 = 0$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70872 ARIANE APARECIDA CRUZ SILVA 19HEA108 - 2 apos 18/09, 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:

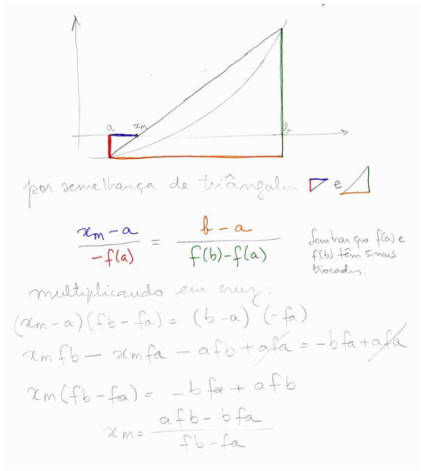
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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:

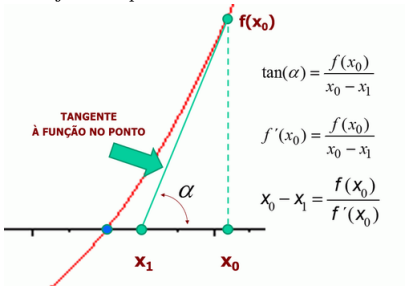
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

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 $x = \sqrt[21]{7402}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 2 * sin(x) + 4 * cos(x) - 4.347804 = 0$ no intervalo entre 0 e 1 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((4 * x)/10) * e^{((7*x)/10)} - 796.774759 = 0$ no intervalo entre 7 e 8 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 4 * x^2 - 50871.636992 = 0$ no intervalo entre 4 e 5 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 71170 ERICK BUENO C DOS SANTOS 19HEA108 - 3 apos 18/09, 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:

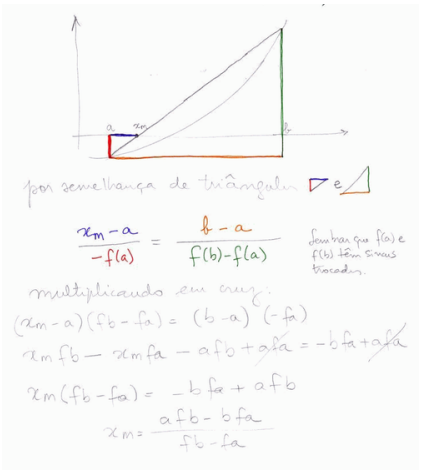
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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:

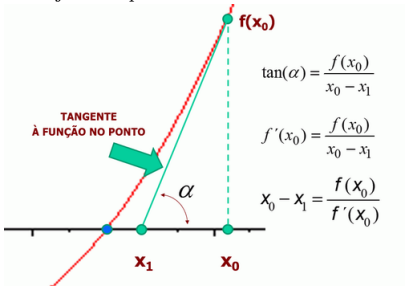
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

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 $x = \sqrt[16]{6579}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 3 * sin(x) + 6 * cos(x) + 4.698803 = 0$ no intervalo entre 4 e 5 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((2 * x)/10) * e^{((8*x)/10)} - 66.634384 = 0$ no intervalo entre 5 e 6 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 5 * x^2 - 15302.276352 = 0$ no intervalo entre 3 e 4 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70889 FABIANA ANDREA SILVA MORALES 19HEA108 - 4 apos 18/09, 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:

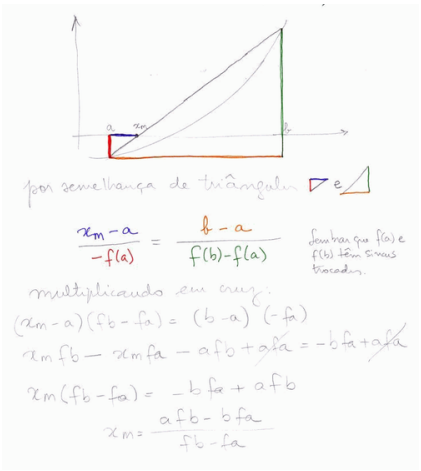
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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:

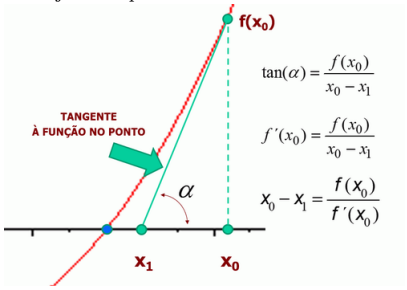
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

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 $x = \sqrt[7]{8912}$ no intervalo entre 3 e 4 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 2 * sin(x) + 7 * cos(x) - 6.518554 = 0$ no intervalo entre 6 e 7 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((8 * x)/10) * e^{((4*x)/10)} - 322.972163 = 0$ no intervalo entre 9 e 10 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 6 * x^5 + 3 * x^2 - 464539.312500 = 0$ no intervalo entre 9 e 10 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70896 FELIPE BORTOLLETTO CIVITATE 19HEA108 - 5 apos 18/09, 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:

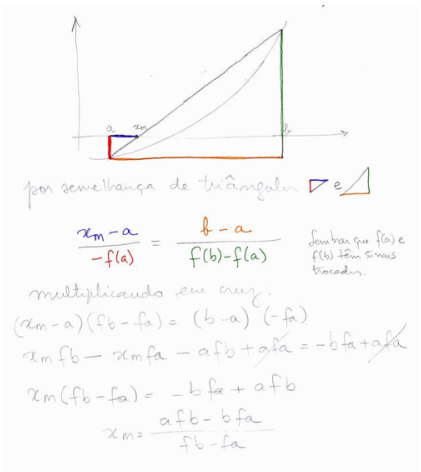
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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

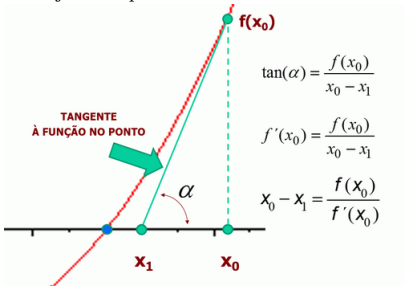
Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
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 $x = \sqrt[18]{6130}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 6 * sin(x) + 7 * cos(x) + 8.969553 = 0$ no intervalo entre 9 e 10 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((8 * x)/10) * e^{((4*x)/10)} - 4.617094 = 0$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 4 * x^2 - 618.002847 = 0$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 71187 GABRIEL CHIMALESKI COSTA 19HEA108 - 6 apos 18/09, 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:

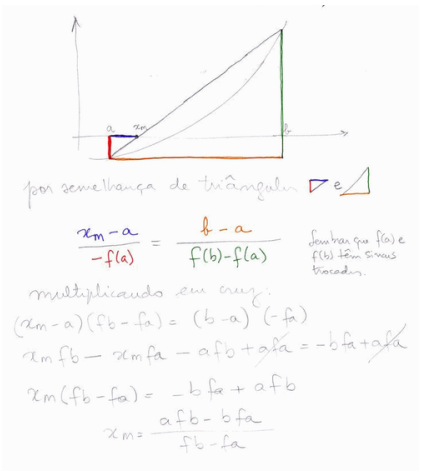
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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

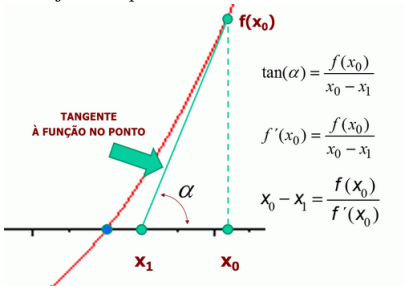
Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
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 $x = \sqrt[22]{5376}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 3 * sin(x) + 2 * cos(x) - 3.344599 = 0$ no intervalo entre 0 e 1 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((6 * x)/10) * e^{((4*x)/10)} - 5.148955 = 0$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 6 * x^4 + 3 * x^2 - 340.734600 = 0$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

108-71187 - 18/09

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70908 GABRIELA MONTEIRO XAVIER 19HEA108 - 7 apos 18/09, 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:

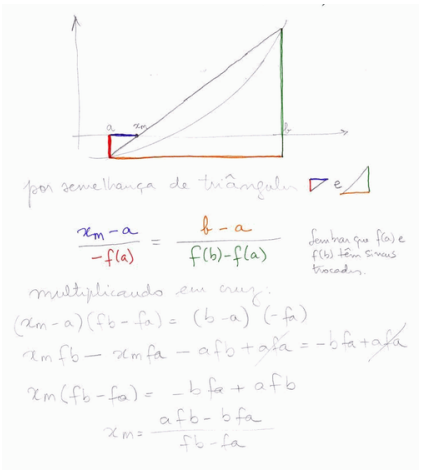
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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:

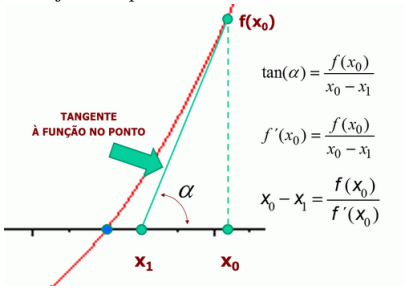
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

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 $x = \sqrt[14]{9465}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 6 * sin(x) + 5 * cos(x) + 1.956081 = 0$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((2 * x)/10) * e^{((8*x)/10)} - 1526.340396 = 0$ no intervalo entre 8 e 9 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 4 * x^3 - 75869.799303 = 0$ no intervalo entre 4 e 5 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70915 GRENDA IZABELI MENEZES SILVA 19HEA108 - 8 apos 18/09, 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:

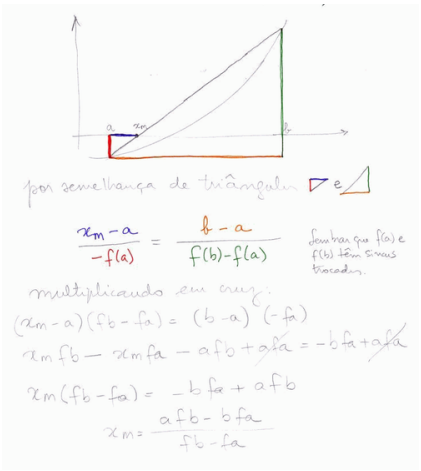
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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

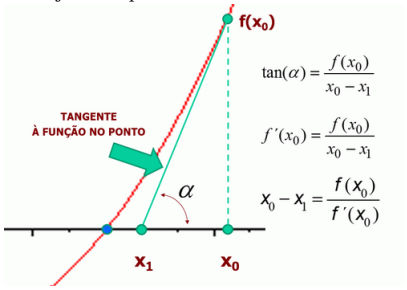
Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
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 $x = \sqrt[26]{5252}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 6 * sin(x) + 7 * cos(x) + 8.711859 = 0$ no intervalo entre 9 e 10 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((3 * x)/10) * e^{((6*x)/10)} - 48.365841 = 0$ no intervalo entre 5 e 6 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 6 * x^5 + 3 * x^2 - .030060 = 0$ no intervalo entre 0 e 1 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70922 HALLYPHER GUERREIRO ZOLLNER 19HEA108 - 9 apos 18/09, 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:

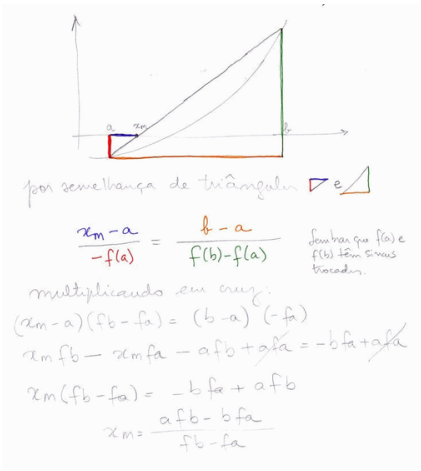
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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:

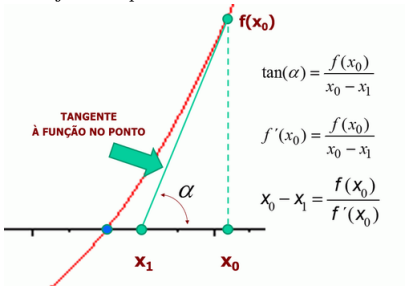
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

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 $x = \sqrt[8]{7879}$ no intervalo entre 3 e 4 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 8 * sin(x) + 2 * cos(x) + 5.549391 = 0$ no intervalo entre 5 e 6 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((8 * x)/10) * e^{((5*x)/10)} - 239.448042 = 0$ no intervalo entre 7 e 8 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 6 * x^4 + 3 * x^2 - 17198.814600 = 0$ no intervalo entre 7 e 8 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70939 HIAN DA SILVA PINTO 19HEA108 - 10 apos 18/09, 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:

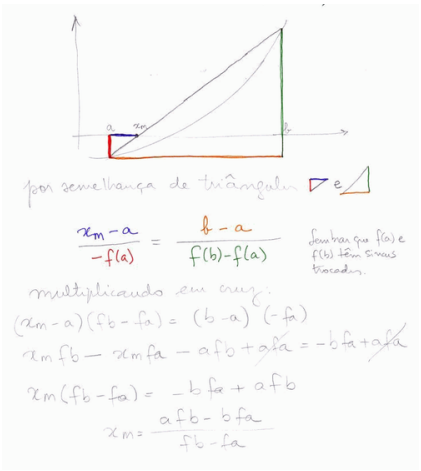
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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:

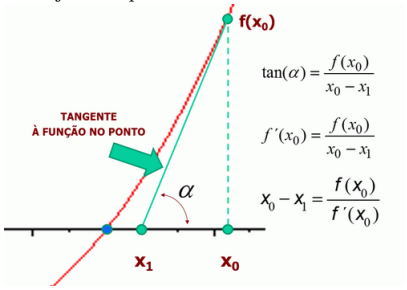
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

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 $x = \sqrt[20]{7370}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 5 * sin(x) + 7 * cos(x) + 5.038559 = 0$ no intervalo entre 9 e 10 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((3 * x)/10) * e^{((6*x)/10)} - 793.724866 = 0$ no intervalo entre 9 e 10 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 6 * x^5 + 4 * x^2 - 12442.418560 = 0$ no intervalo entre 4 e 5 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70946 IGOR HERNAN ANDRADE SANTOS 19HEA108 - 11 apos 18/09, 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:

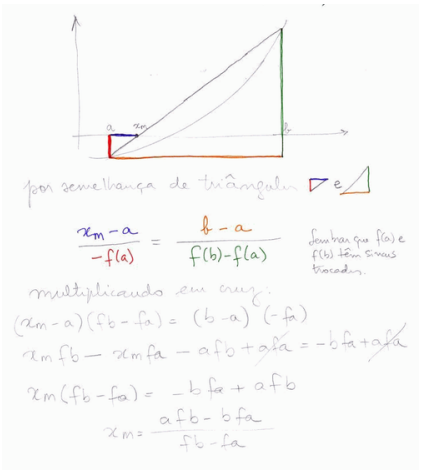
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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:

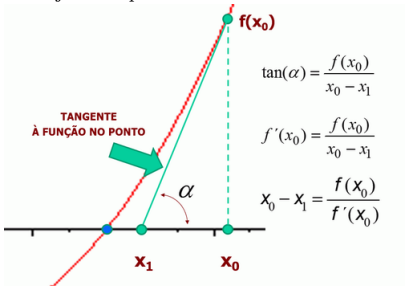
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

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 $x = \sqrt[26]{8445}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 5 * sin(x) + 6 * cos(x) + 6.358789 = 0$ no intervalo entre 9 e 10 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((5 * x)/10) * e^{((8*x)/10)} - 1660.710940 = 0$ no intervalo entre 7 e 8 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 5 * x^2 - 633433.125183 = 0$ no intervalo entre 6 e 7 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

108-70946 - 18/09

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70953 ISADORA FONSECA CARRARI 19HEA108 - 12 apos 18/09, 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:

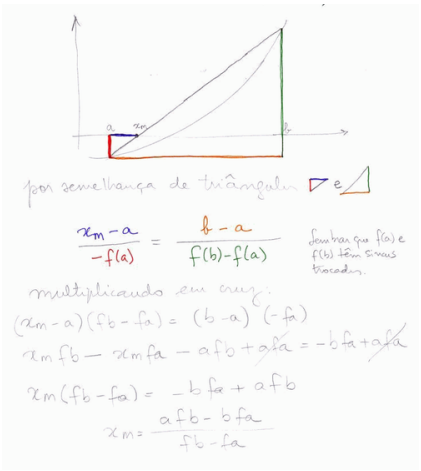
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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

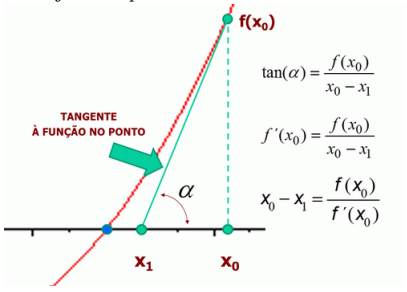
Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
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 $x = \sqrt[12]{8022}$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 8 * sin(x) + 3 * cos(x) - 2.324911 = 0$ no intervalo entre 6 e 7 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((6 * x)/10) * e^{((8*x)/10)} - 4177.240258 = 0$ no intervalo entre 8 e 9 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^5 + 4 * x^2 - 64283.058240 = 0$ no intervalo entre 6 e 7 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70960 JESSICA FURTUOSO HARMATIUK 19HEA108 - 13 apos 18/09, 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:

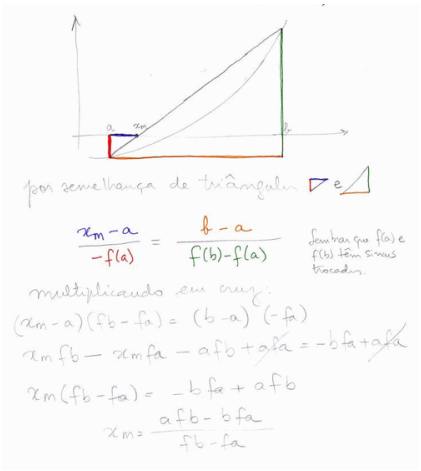
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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

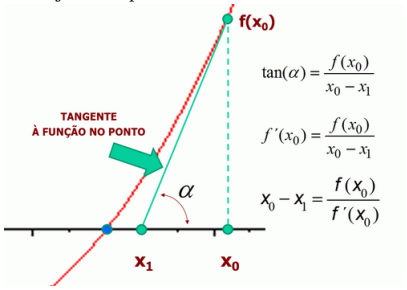
Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
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 $x = \sqrt[20]{7292}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 4 * sin(x) + 3 * cos(x) - 4.760934 = 0$ no intervalo entre 6 e 7 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((7 * x)/10) * e^{((4*x)/10)} - 21.561031 = 0$ no intervalo entre 4 e 5 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 6 * x^5 + 4 * x^2 - 464629.562500 = 0$ no intervalo entre 9 e 10 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70977 JOAO GUILHERME LIMA MARIANO 19HEA108 - 14 apos 18/09, 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:

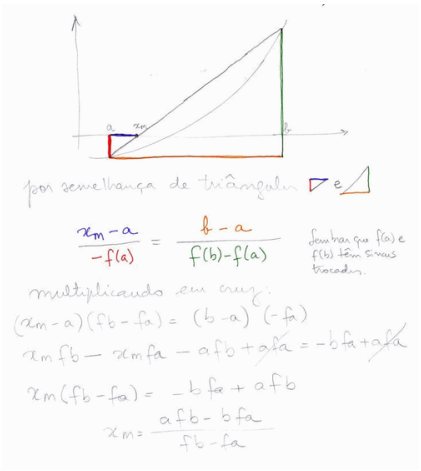
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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:

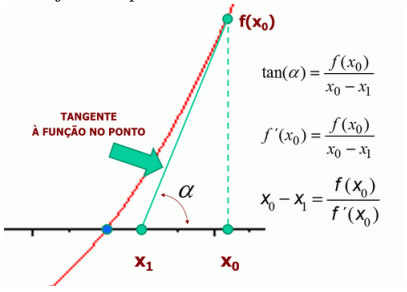
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

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 $x = \sqrt[14]{5020}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 5 * sin(x) + 2 * cos(x) + 4.890695 = 0$ no intervalo entre 3 e 4 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((4 * x)/10) * e^{((8*x)/10)} - 28.560998 = 0$ no intervalo entre 3 e 4 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 5 * x^2 - 1059606.034023 = 0$ no intervalo entre 7 e 8 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

108-70977 - 18/09

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70984 KAREN LETICIA SARTORI 19HEA108 - 15 apos 18/09, 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:

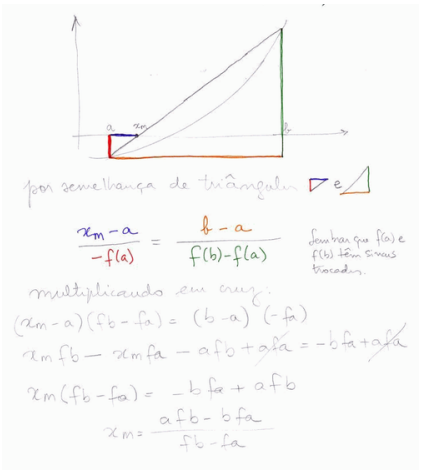
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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

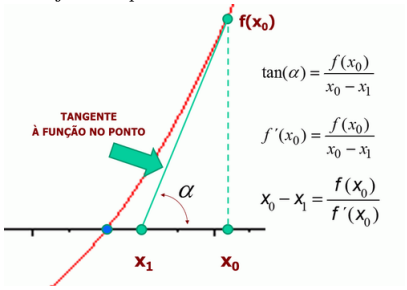
Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
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 $x = \sqrt[27]{6116}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 7 * sin(x) + 8 * cos(x) - 9.423136 = 0$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((2 * x)/10) * e^{((8*x)/10)} - .735596 = 0$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^5 + 4 * x^3 - .004070 = 0$ no intervalo entre 0 e 1 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 70991 KARLA KOSOSKI BARTH MACEDO 19HEA108 - 16 apos 18/09, 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:

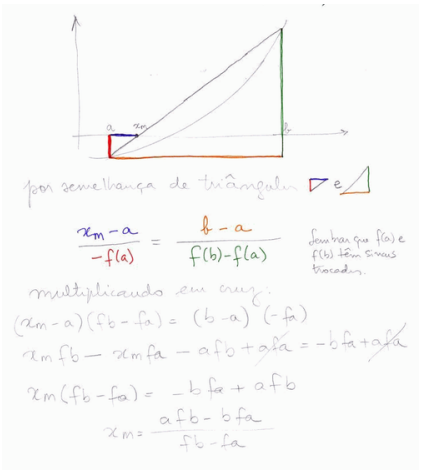
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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

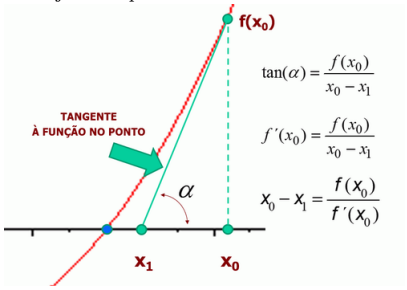
Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
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 $x = \sqrt[27]{8418}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 3 * sin(x) + 5 * cos(x) + 4.205548 = 0$ no intervalo entre 9 e 10 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((3 * x)/10) * e^{((4*x)/10)} - 38.478830 = 0$ no intervalo entre 7 e 8 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 5 * x^2 - .050007 = 0$ no intervalo entre 0 e 1 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 71006 LETICIA ERTHAL CORDEIRO 19HEA108 - 17 apos 18/09, 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:

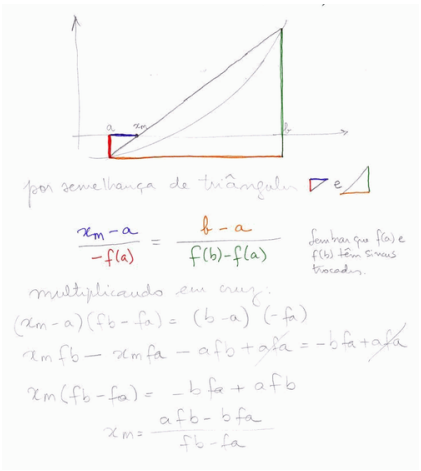
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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:

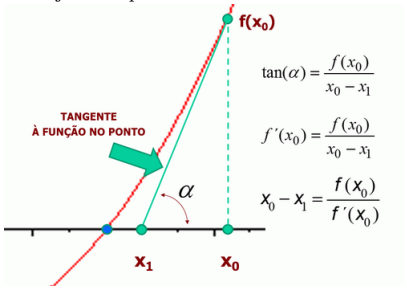
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

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 $x = \sqrt[22]{6800}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 6 * sin(x) + 2 * cos(x) - 6.269343 = 0$ no intervalo entre 7 e 8 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((5 * x)/10) * e^{((4*x)/10)} - 83.769849 = 0$ no intervalo entre 7 e 8 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 4 * x^2 - 240205.090743 = 0$ no intervalo entre 5 e 6 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 71013 LIAH CHIURATTO
BUBA 19HEA108 - 18 apos 18/09, 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:

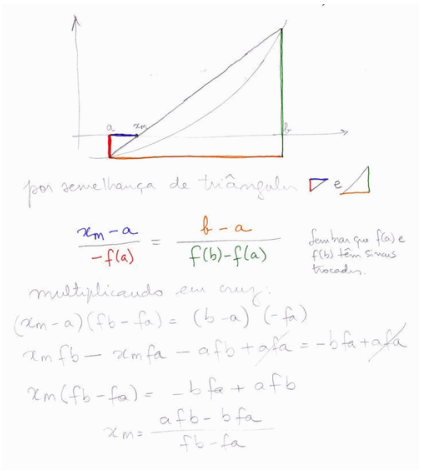
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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:

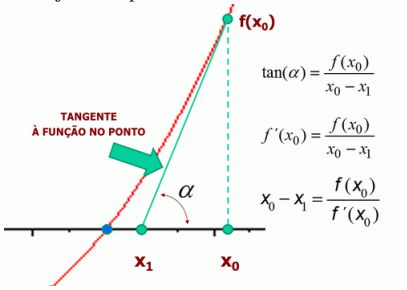
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

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 $x = \sqrt[21]{9988}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 4 * sin(x) + 2 * cos(x) - 4.471927 = 0$ no intervalo entre 7 e 8 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((5 * x)/10) * e^{((4*x)/10)} - 16.370300 = 0$ no intervalo entre 4 e 5 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 4 * x^2 - 2832266.485952 = 0$ no intervalo entre 8 e 9 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 71020 LISA ISABELLA FISCHER 19HEA108 - 19 apos 18/09, 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:

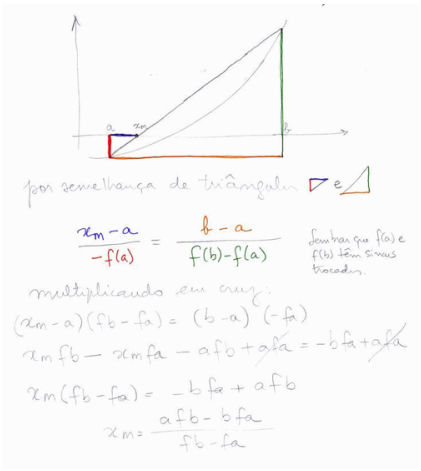
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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:

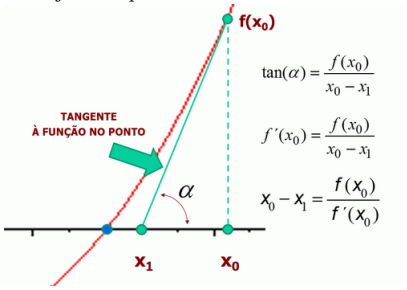
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

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 $x = \sqrt[8]{5208}$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 8 * sin(x) + 5 * cos(x) + 5.458467 = 0$ no intervalo entre 3 e 4 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((5 * x)/10) * e^{((7*x)/10)} - 39.723277 = 0$ no intervalo entre 4 e 5 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^5 + 4 * x^3 - 66.656250 = 0$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 71037 LUIS GABRIEL COUTINHO ROCHA F 19HEA108 - 20 apos 18/09, 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:

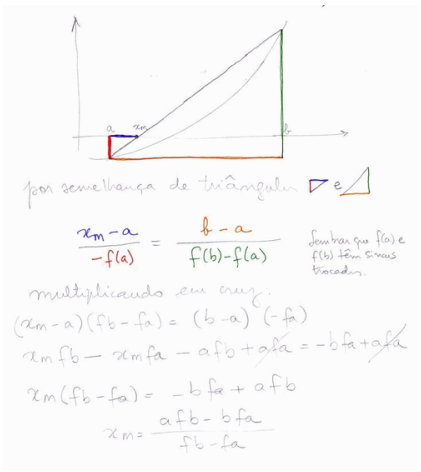
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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

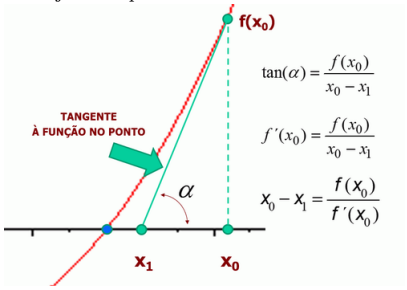
Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
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 $x = \sqrt[21]{5825}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 6 * sin(x) + 8 * cos(x) - 2.416927 = 0$ no intervalo entre 5 e 6 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((5 * x)/10) * e^{((3*x)/10)} - 7.810491 = 0$ no intervalo entre 4 e 5 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^5 + 4 * x^2 - 166338.281250 = 0$ no intervalo entre 7 e 8 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 71044 MARIA GABRIELA FERNANDES LOPES 19HEA108 - 21 apos 18/09, 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:

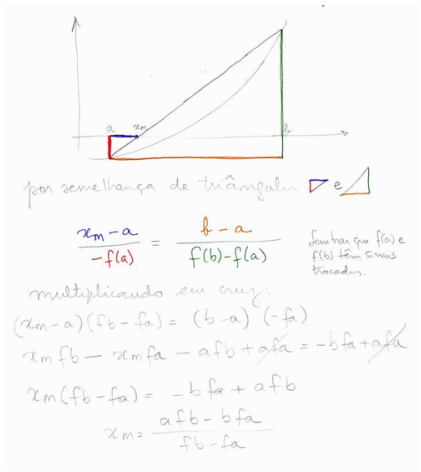
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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

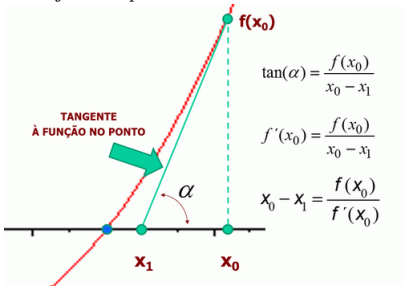
Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
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 $x = \sqrt[17]{6854}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 4 * sin(x) + 3 * cos(x) - 4.362549 = 0$ no intervalo entre 6 e 7 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((2 * x)/10) * e^{((3*x)/10)} - 10.000845 = 0$ no intervalo entre 6 e 7 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^5 + 3 * x^2 - 541917.406250 = 0$ no intervalo entre 9 e 10 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 71051 MARIA LUIZA MOCELIM DE MATTOS 19HEA108 - 22 apos 18/09, 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:

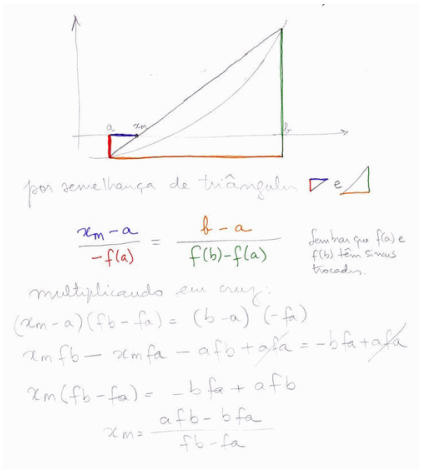
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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:

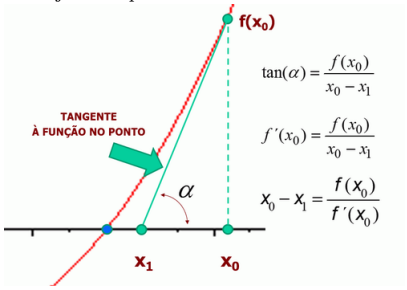
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

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 $x = \sqrt[27]{7332}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 5 * sin(x) + 7 * cos(x) + 8.115153 = 0$ no intervalo entre 4 e 5 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((6 * x)/10) * e^{((3*x)/10)} - 18.028268 = 0$ no intervalo entre 5 e 6 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^5 + 4 * x^2 - 215643.587930 = 0$ no intervalo entre 7 e 8 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 71068 MATEUS DOS SANTOS NEVES 19HEA108 - 23 apos 18/09, 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:

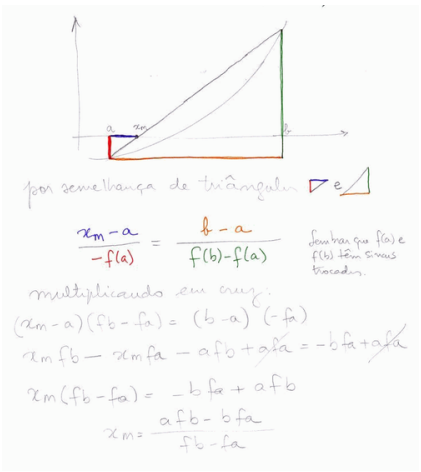
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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:

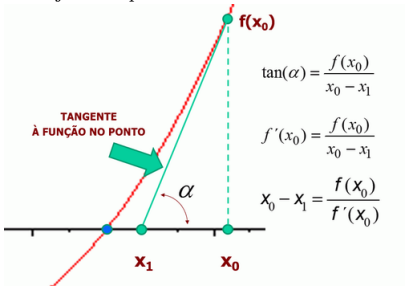
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

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 $x = \sqrt[17]{9866}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 3 * sin(x) + 6 * cos(x) - 4.443241 = 0$ no intervalo entre 5 e 6 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((5 * x)/10) * e^{((7*x)/10)} - 307.555327 = 0$ no intervalo entre 6 e 7 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 6 * x^5 + 4 * x^2 - 262.686060 = 0$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 71075 PAOLA GEBAUER
MATEOS 19HEA108 - 24 apos 18/09, 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:

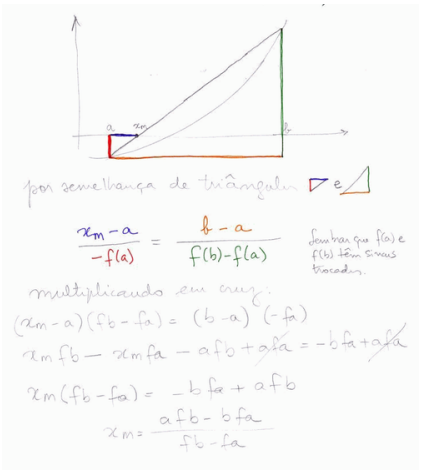
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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

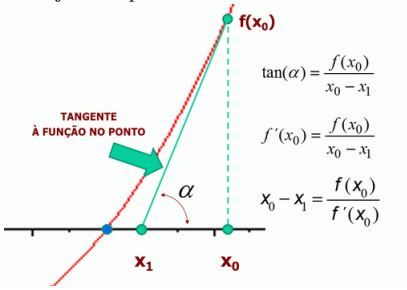
Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
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 $x = \sqrt[9]{5340}$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 4 * sin(x) + 2 * cos(x) + 3.608526 = 0$ no intervalo entre 9 e 10 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((7 * x)/10) * e^{((3*x)/10)} - 26.618696 = 0$ no intervalo entre 6 e 7 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 4 * x^3 - 44567.569343 = 0$ no intervalo entre 4 e 5 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 71101 RAUL BECKERT
VESSARO 19HEA108 - 25 apos 18/09, 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:

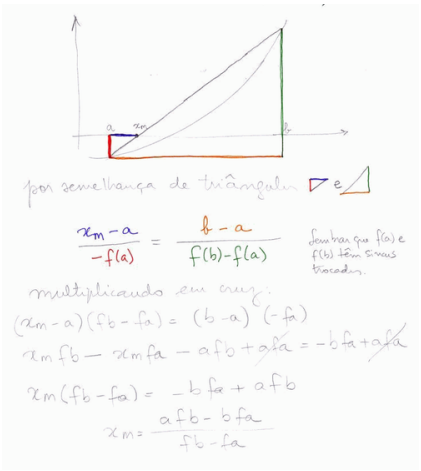
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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

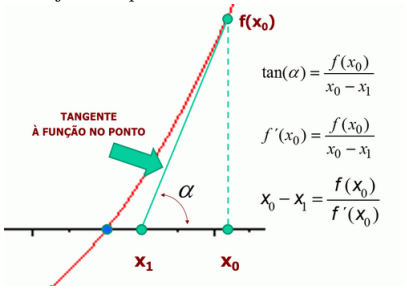
Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
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 $x = \sqrt[8]{7124}$ no intervalo entre 3 e 4 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 5 * sin(x) + 4 * cos(x) - 5.270424 = 0$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((7 * x)/10) * e^{((4*x)/10)} - 123.649233 = 0$ no intervalo entre 7 e 8 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^6 + 5 * x^4 - 4009372.244787 = 0$ no intervalo entre 9 e 10 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 71118 VICTORIA FERRO MAESTRELLI 19HEA108 - 26 apos 18/09, 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:

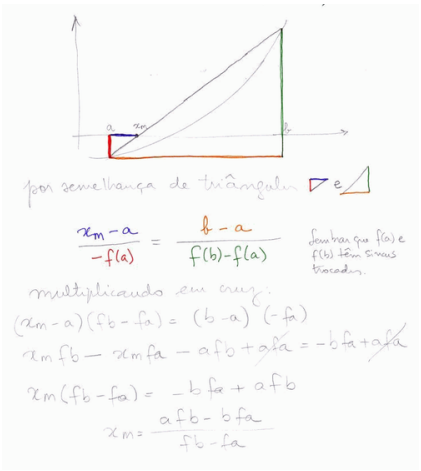
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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

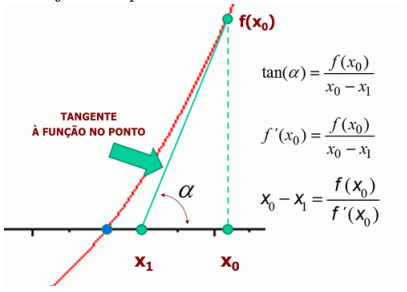
Veja a seguir a solução da mesma equação, $(e^x - sen(x) - 2 = 0)$ com os mesmos intervalos do método anterior.

```
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 $x = \sqrt[14]{9196}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 2 * sin(x) + 6 * cos(x) + 5.085498 = 0$ no intervalo entre 4 e 5 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((7 * x)/10) * e^{((4*x)/10)} - 12.162119 = 0$ no intervalo entre 3 e 4 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 6 * x^4 + 3 * x^2 - 12225.342600 = 0$ no intervalo entre 6 e 7 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

UFPR - UP - UTFPR - PUCPr – 24/08/2019 - 11:32:41.6
Prof Dr P Kantek (pkantek@gmail.com)
Cálculo Numérico - Solução de equações
VIVO734a, V: 1.55 71125 VITOR KURTEN
FEITOSA 19HEA108 - 27 apos 18/09, 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:

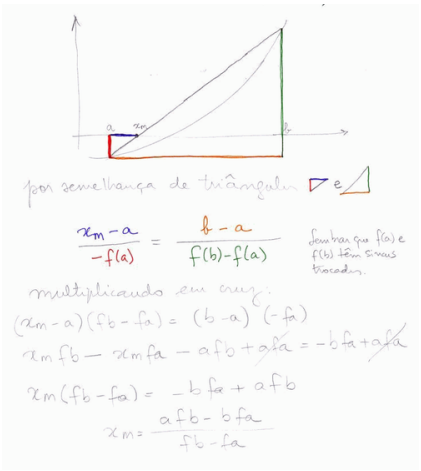
1. Arbitrar intervalo em $[a, b]$ que compreenda a raiz.
2. Arbitrar um erro permitido ϵ
3. Calcular $x_m = (a + b)/2$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $b - a > \epsilon$ voltar ao passo 3.

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:

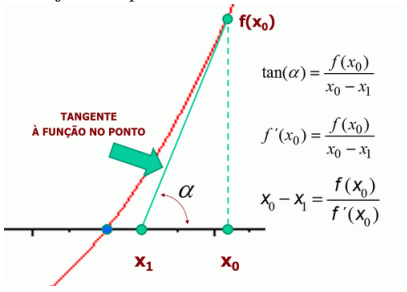
1. Arbitrar um intervalo $[a, b]$ que compreende a raiz
2. Arbitrar ϵ
3. Calcular $x_m = \frac{a.f(b) - b.f(a)}{f(b) - f(a)}$
4. Se $f(a).f(x_m) < 0$ o novo intervalo será $[a, x_m]$
5. Se $f(a).f(x_m) > 0$ o novo intervalo será $[x_m, b]$
6. Enquanto o erro $f(x_m) > \epsilon$ voltar ao passo 3

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 $x = \sqrt[20]{9755}$ no intervalo entre 1 e 2 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

2. Idem $y = 5 * sin(x) + 7 * cos(x) - 8.463727 = 0$ no intervalo entre 0 e 1 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

3. Idem $y = ((3 * x)/10) * e^{((5*x)/10)} - 19.594869 = 0$ no intervalo entre 5 e 6 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	

4. Idem $y = 7 * x^5 + 3 * x^2 - 1461.010430 = 0$ no intervalo entre 2 e 3 e responda a seguir os 3 valores encontrados

bissecção	
cordas	
Newton	