C++ Finite-Differences and Black-Scholes Equation

Issues with the Greeks in simple schemes
Step 1:
Compare C++ coded Greeks with output of symbolic differentiation in Mathematica - ensures C++ formulae solid.

Case Study: the Vanilla European Put Option

Black-Scholes Model for Verification of C++ Greeks

```
Off[General::spell1]
```

From last time - nothing new here:

```
Ncdf[(z_)?NumberQ] := N[0.5*Erf[z/Sqrt[2]] + 0.5];
Ncdf[x_] := (1 + Erf[x/Sqrt[2]])/2;
done[s_, σ_, k_, t_, r_, q_] :=
((r - q)*t + Log[s/k])/(σ*Sqrt[t]) + (σ*Sqrt[t])/2;
dtwo[s_, σ_, k_, t_, r_, q_] :=
((r - q)*t + Log[s/k])/(σ*Sqrt[t]) - (σ*Sqrt[t])/2;
```

```
BlackScholesCall[s_, k_, σ_, r_, q_, t_] :=
s*Exp[-q*t]*Ncdf[done[s, σ, k, t, r, q]] - k*Exp[-r*t]*Ncdf[dtwo[s, σ,
BlackScholesPut[s_, k_, σ_, r_, q_, t_] :=
k*Exp[-r*t]*Ncdf[-dtwo[s, σ, k, t, r, q]] - s*Exp[-q*t]*Ncdf[-done[s, σ
```

Build Greeks automatically for verification

In *Mathematica* we can just calculcate the derivatives automatically

```
BlackScholesCallDelta[s_,k_, v_, r_, q_, t_]=
Evaluate[Simplify[D[BlackScholesCall[s,k, v, r, q, t], s]]];

BlackScholesPutDelta[s_,k_, v_, r_, q_, t_]=
Evaluate[Simplify[D[BlackScholesPut[s,k, v, r, q, t], s]]];

BlackScholesCallGamma[s_,k_, v_, r_, q_, t_]=
Evaluate[D[BlackScholesCall[s,k, v, r, q, t], {s, 2}]];

BlackScholesPutGamma[s_,k_, v_, r_, q_, t_]=
Evaluate[D[BlackScholesPut[s,k, v, r, q, t], {s, 2}]];

BlackScholesCallTheta[s_,k_, v_, r_, q_, t_]=
-Evaluate[D[BlackScholesCall[s,k, v, r, q, t], t]];

BlackScholesPutTheta[s_,k_, v_, r_, q_, t_]=
-Evaluate[D[BlackScholesPut[s,k, v, r, q, t], t]];

BlackScholesCallRho[s_,k_, v_, r_, q_, t_]=
Evaluate[D[BlackScholesCall[s,k, v, r, q, t], r]];

BlackScholesPutRho[s_,k_, v_, r_, q_, t_]=
Evaluate[D[BlackScholesPut[s,k, v, r, q, t], r]];

BlackScholesCallVega[s_,k_, v_, r_, q_, t_]=
Evaluate[D[BlackScholesCall[s,k, v, r, q, t], v]];

BlackScholesPutVega[s_,k_, v_, r_, q_, t_]=
Evaluate[D[BlackScholesPut[s,k, v, r, q, t], v]];
```

To the C++ model

Run

blackscholesgreeks.cpp

with test parameters:

N+- = 100;

range = +-50;

K=100;

σ=0.2;

r=0.1;

q=0.05;

t=2

```
SetDirectory[
 "C:\Documents and Settings\William Shaw\My
   Documents\LGS\cppexamples"]
```

```
C:\Documents and Settings\William
  Shaw\My Documents\LGS\cppexamples
```

```
FileNames[]
```

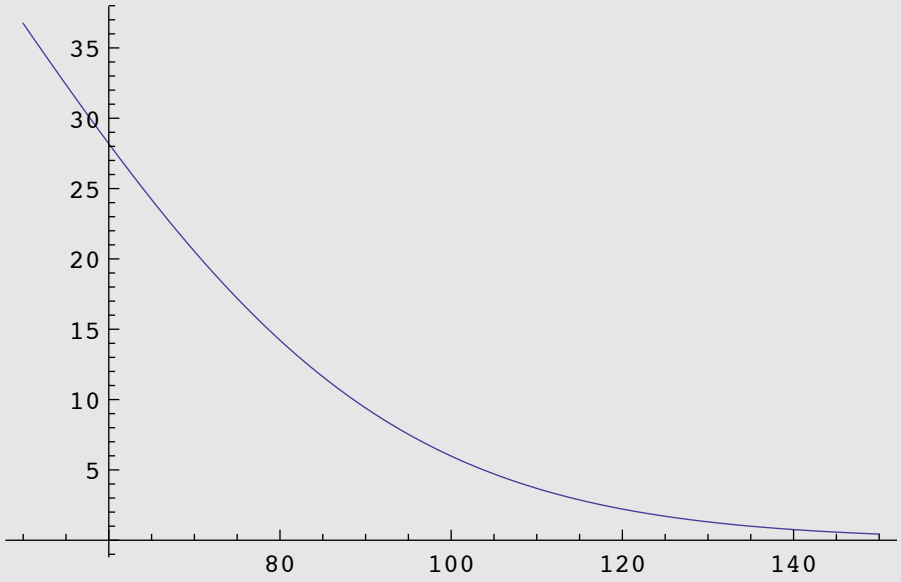
```
{arithmeticone.cpp, arithmeticone.exe,
 arithmeticonefile.cpp, arithmeticonefile.exe,
 blackscholes.cpp, blackscholesgreeks.cpp,
 blackscholesgreeks_cpp.txt, blackscholesgreeks.exe,
 bsfindiffvsanalytic_cpp.txt, bsgreeksoutput.txt,
 cumulativenormals.cpp, cumulativenormals.exe,
 dowhileiteration.cpp, dowhileiteration.exe,
 foriteration.cpp, foriteration.exe, hello.cpp,
 hello.exe, helloname.cpp, helloname.exe, ifdeciding.cpp,
 ifdeciding.exe, incrementing.cpp, incrementing.exe,
 myoutput.txt, normaloutput.txt, realtypes.cpp,
 realtypes.exe, switchdeciding.cpp, switchdeciding.exe,
 whileiteration.cpp, whileiterationcruder.cpp,
 whileiterationcruder.exe, whileiteration.exe}
```

```
cputdata = ReadList["bsgreeksoutput.txt", Number];
tabular = Partition[cputdata, 7];
len = Length[tabular]
```

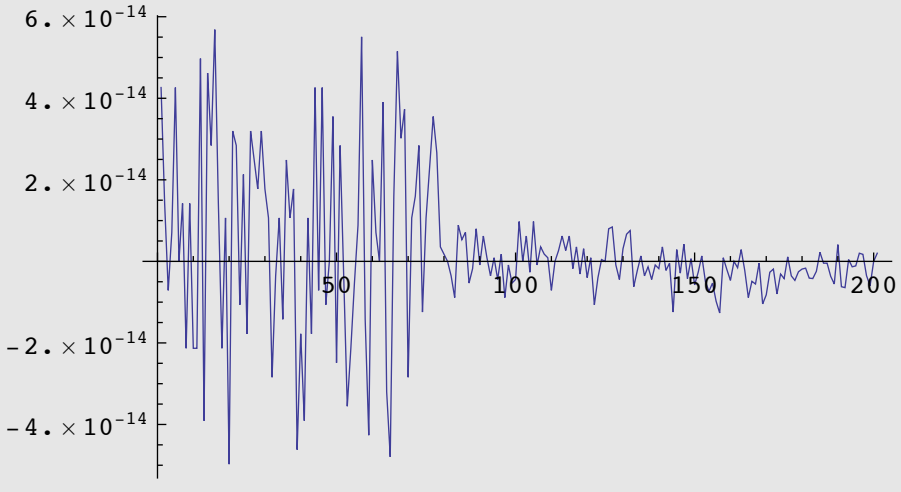
```
201
```

Compare Values

```
valdata = Table[{tabular[[i, 1]], tabular[[i, 2]]}, {i, 1, len}];
ListPlot[valdata, Joined → True]
```

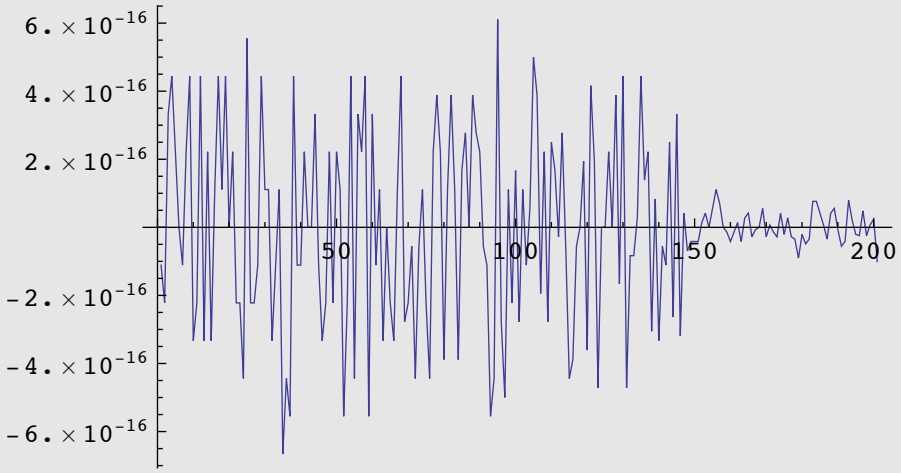



```
comparevaldata =
  Table[BlackScholesPut[valdata[[i, 1]], 100, 0.2`, 0.1`,
     0.05`, 2] - tabular[[i, 2]], {i, 1, len}];
ListPlot[comparevaldata, Joined → True]
```



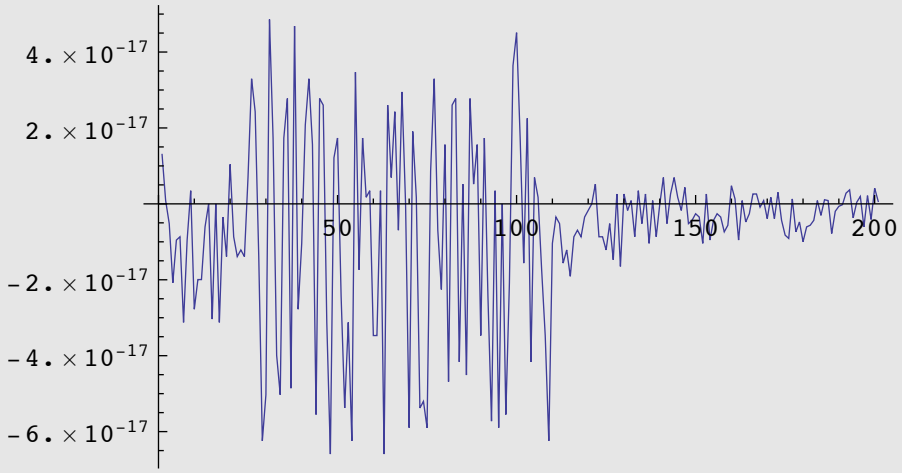
Compare Deltas

```
comparedeltadata =
  Table[BlackScholesPutDelta[valdata[[i, 1]], 100, 0.2`,
     0.1`, 0.05`, 2] - tabular[[i, 3]], {i, 1, len}];
ListPlot[comparedeltadata, Joined → True]
```



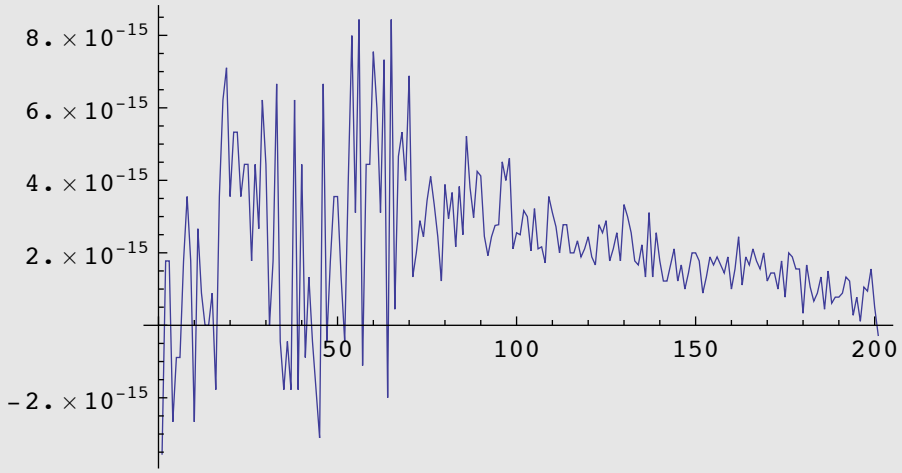
Compare Gammas

```
comparegammadata =
  Table[BlackScholesPutGamma[valdata[[i, 1]], 100, 0.2`,
     0.1`, 0.05`, 2] - tabular[[i, 4]], {i, 1, len}];
ListPlot[comparegammadata, Joined → True]
```



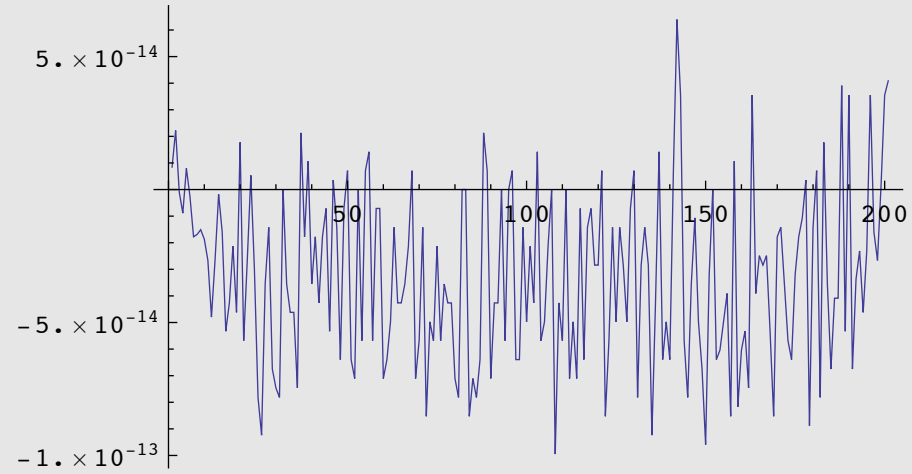
Compare Thetas

```
comparethetadata =
  Table[BlackScholesPutTheta[valdata[[i, 1]], 100, 0.2`,
     0.1`, 0.05`, 2] - tabular[[i, 5]], {i, 1, len}];
ListPlot[comparethetadata, Joined → True]
```



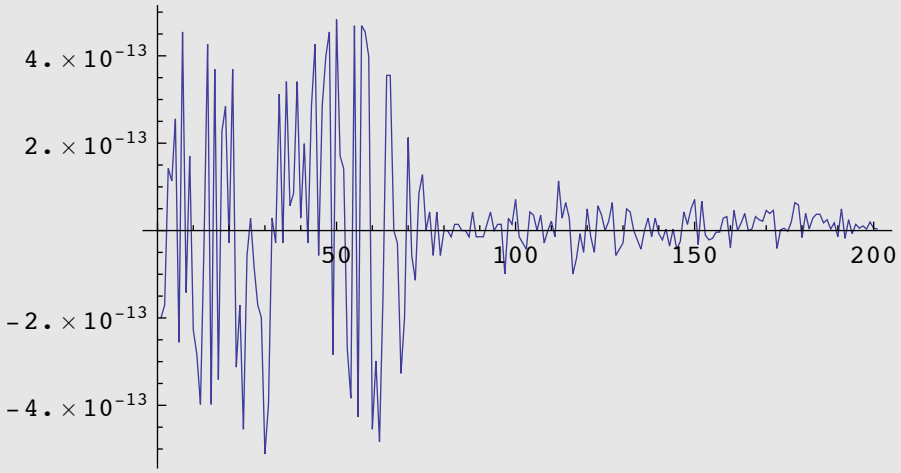
Compare Vegas

```
comparevegadata =
  Table[BlackScholesPutVega[valdata[[i, 1]], 100, 0.2`,
     0.1`, 0.05`, 2] - tabular[[i, 6]], {i, 1, len}];
ListPlot[comparevegadata, Joined → True]
```



Compare Rhos

```
comparerhodata =
  Table[BlackScholesPutRho[valdata[[i, 1]], 100, 0.2`,
     0.1`, 0.05`, 2] - tabular[[i, 7]], {i, 1, len}];
ListPlot[comparerhodata, Joined → True, PlotRange → All]
```



You can do other things like check BS equation; vega-gamma and rho-delta identities.

$$\theta + (r S - q)\,\Delta + \frac{1}{2}\,\sigma^2 S^2\,\Gamma - r V = 0 \tag{1}$$

$$\Lambda = S^2\,\sigma\Gamma(T - t) \tag{2}$$

$$\rho = -(T - t)\,(V - S\,\Delta) \tag{3}$$

These are always valid for simple European options. Better used as a check rather than wired into code (if you do that rather than finding a bug you might multiply it!) unless you need to use them for numerical efficiency, e.g. in developing a C++ code that makes just ONE run to do all the Greeks (for Euros only, for now).

Finite Differences: Values and Greeks all in C++

Run C++ code bsfindiffvsanalytic.cpp with the parameters:

N=160;

M=20;

dx = 0.025;

dτ = 0.005;

K=10;

σ = 0.2

r=0.05;

q = 0.0;

T-t = 5.0

θ = 0.5 (for CN scheme)

```
FileNames["*.txt"]
```

```
{blackscholesgreeks_cpp.txt,
 bsfdvsanal.txt, bsfindiffvsanalytic_cpp.txt,
 bsgreeksoutput.txt, myoutput.txt, normaloutput.txt}
```

```
cputtest = ReadList["bsfdvsanal.txt", Number];
tabular = Partition[cputtest, 5];
len = Length[tabular]
```

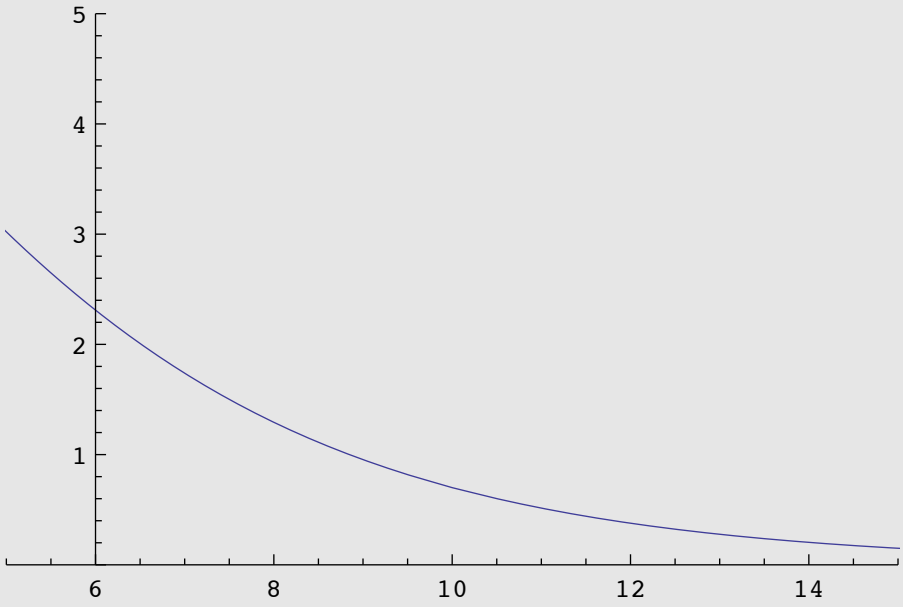
```
321
```

Compare Valuations

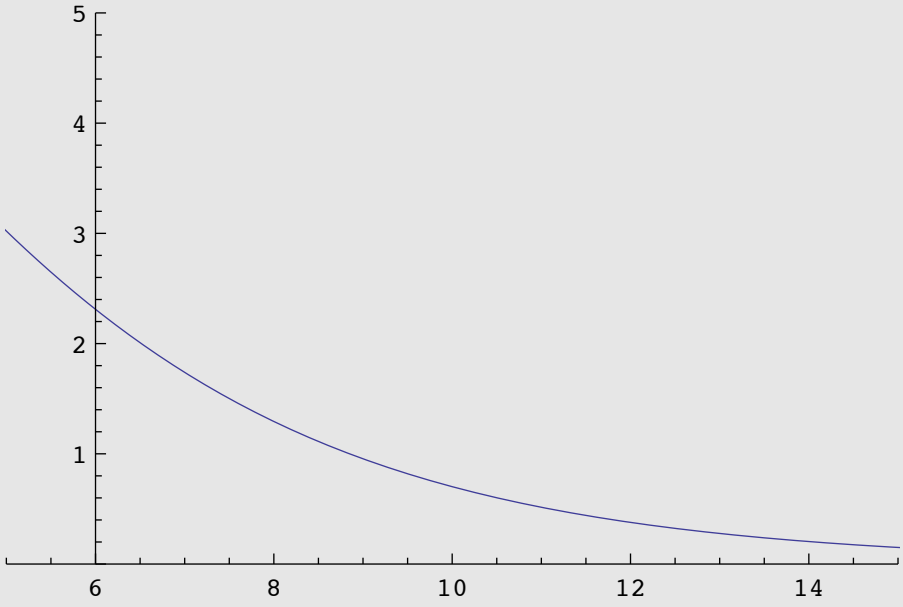
results are of the form of rows with

{S, FD Val, Analytical VAL, FD theta, Analytical Theta} in entries 1 to 5.

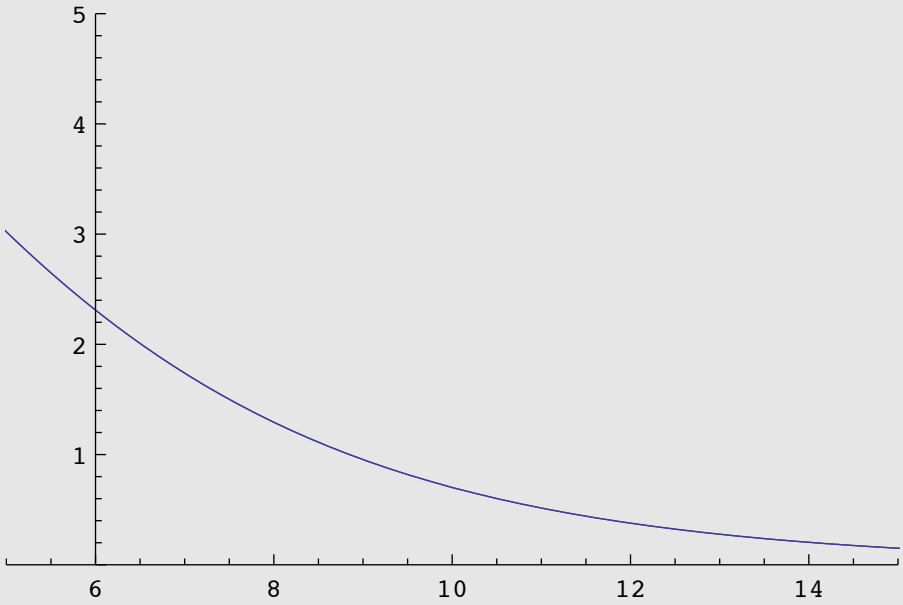
```
comparevaldataa = Table[{tabular[[i, 1]], tabular[[i, 2]]},
    {i, 1, len}];
valplota = ListPlot[comparevaldataa, Joined → True,
  PlotRange → {{5, 15}, {0, 5}}]
```



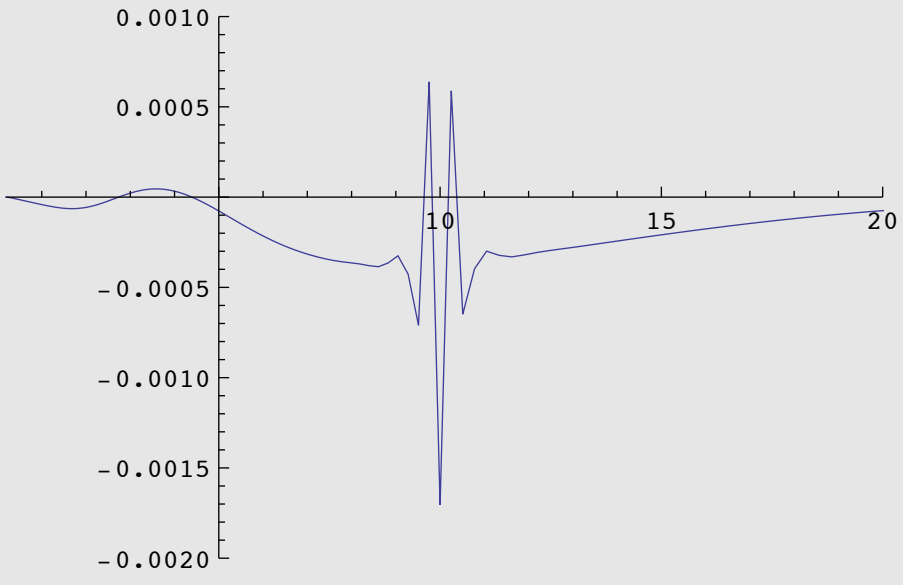

```
comparevaldatab = Table[{tabular[[i, 1]], tabular[[i, 3]]},
    {i, 1, len}];
valplotb = ListPlot[comparevaldatab, Joined → True,
  PlotRange → {{5, 15}, {0, 5}}]
```



```
Show[valplota, valplotb]
```

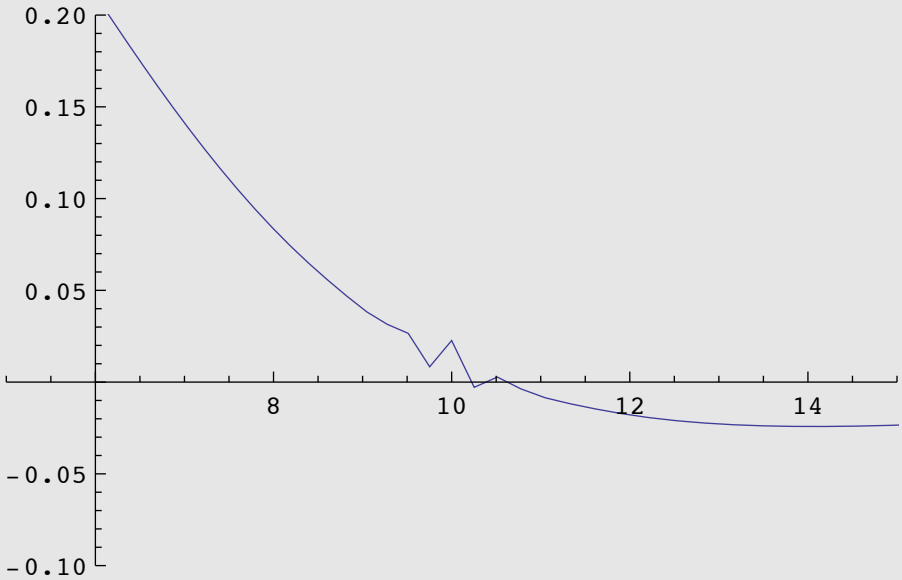



```
valerrordata =
  Table[{tabular[[i, 1]], tabular[[i, 2]] - tabular[[i, 3]]},
   {i, 1, len}];
errorplot = ListPlot[valerrordata, Joined → True,
  PlotRange → {{0.2`, 20}, {-0.002`, 0.001`}}]
```

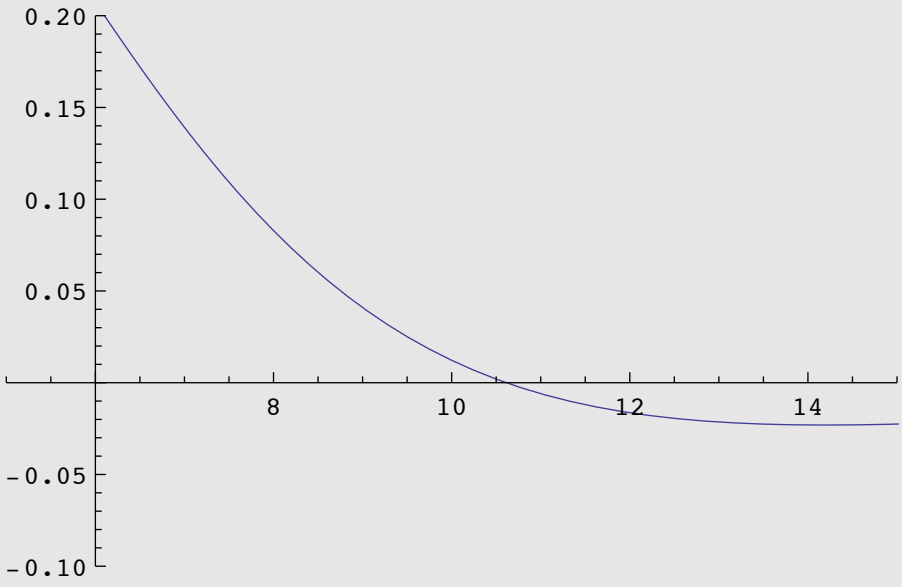


Compare Thetas

```
comparethetadataa = Table[{tabular[[i, 1]], tabular[[i, 4]]},
    {i, 1, len}];
thetaplota = ListPlot[comparethetadataa, Joined → True,
  PlotRange → {{5, 15}, {-0.1`, 0.2`}}]
```



```
comparethetadatab = Table[{tabular[[i, 1]], tabular[[i, 5]]},
    {i, 1, len}];
thetaplotb = ListPlot[comparethetadatab, Joined → True,
  PlotRange → {{5, 15}, {-0.1`, 0.2`}}]
```




```
Show[thetaplota, thetaplotb]
```