Computing Local Volatility and Implied Volatility

Fitting Implied Volatility Surface

```
> restart :
> with(LinearAlgebra) :
  with(Finance) :
  with(Statistics) :
  with(plots) :
  interface(displayprecision = 5) :
```

First let us import prices of S&P 500 call options available on October 27, 2006.

```
> Data := ImportMatrix("this:///sp500.csv")
```

$$Data := \left[\begin{array}{c} \text{309 x 10 Matrix} \\ \text{Data Type: anything} \\ \text{Storage: rectangular} \\ \text{Order: Fortran\_order} \end{array}\right] \quad (1.1)$$

This data can be stored in a DataFrame.

```
> DF := DataFrame(Data[2 ..,..], columns = Data[1,..])
```

```
DF := [[, "Month", "Year", "Strike", "Symbol", "Last", "Change", "Bid", "Ask", ... ],        (1.2)
    [1, "November", 2006, 1475, "SXZKO", 0.10000, -0.10000, 0.05000, 0.15000, ... ],
    [2, "November", 2006, 1500, "SXMKT", 0.05000, -0.05000, 0.05000, 0.10000, ... ],
    [3, "November", 2006, 1525, "SXMKE", 0.10000, 0.05000, 0.05000, 0.10000, ... ],
    [4, "November", 2006, 1450, "SXZKJ", 0.30000, 0, 0.10000, 0.25000, ... ],
    [5, "November", 2006, 1460, "SXZKL", 0.35000, 0.15000, 0.10000, 0.40000, ... ],
    [6, "November", 2006, 1455, "SXZKK", 0.40000, 0, 0.10000, 0.40000, ... ],
```

[7, "November", 2006, 1440, "SXZKH", 0.50000, 0.30000, 0.15000, 0.50000, ...],
[8, "November", 2006, 1435, "SXZKG", 0.50000, −0.40000, 0.35000, 0.55000, ...],
[..., ..., ..., ..., ..., ..., ..., ..., ..., ...]]

Extract data from this DataFrame.

> $M, N := upperbound(DF)$

$$M, N := 308, 10 \quad \textbf{(1.3)}$$

The data for the "month" data series needs to be converted to names in order to work with the Finance package:

> $DF["Month"] := ((x) \rightarrow convert(x, name)) \sim (DF["Month"])$

$$DF_{"Month"} := \begin{bmatrix} 1 & November \\ 2 & November \\ 3 & November \\ 4 & November \\ 5 & November \\ 6 & November \\ 7 & November \\ ... & ... \end{bmatrix} \quad \textbf{(1.4)}$$

Value of the underlying, risk-free rate and dividend yield.

> $S0 := 1377$

$$S0 := 1377 \quad \textbf{(1.5)}$$

> $r := 0.0532$

$$r := 0.05320 \quad \textbf{(1.6)}$$

> $d := 0.0182$

$$d := 0.01820 \quad \textbf{(1.7)}$$

Extract exercise dates and times for which data is available.

> $ExerciseDates := [seq(NthWeekday(3, Friday, DF[i, "Month"], DF[i, "Year"]), i = 1 ..M)]:$

> $Times := map(t \rightarrow YearFraction("October 27, 2006", t), ExerciseDates):$

Implied volatilities for options maturing in December 2006.

> $Time_1 := YearFraction("October 27, 2006", NthWeekday(3, Friday, December, 2006))$

$$Time_1 := 0.13425 \quad (1.8)$$

Extract a subset of the DataFrame corresponding to the observations in December 2006:

> $Dec2006 := DF[DF["Month"] =\sim December \textbf{ and } DF["Year"] =\sim 2006]$

$$Dec2006 := [[, "Month", "Year", "Strike", "Symbol", "Last", "Change", "Bid", "Ask", ...], \quad (1.9)$$

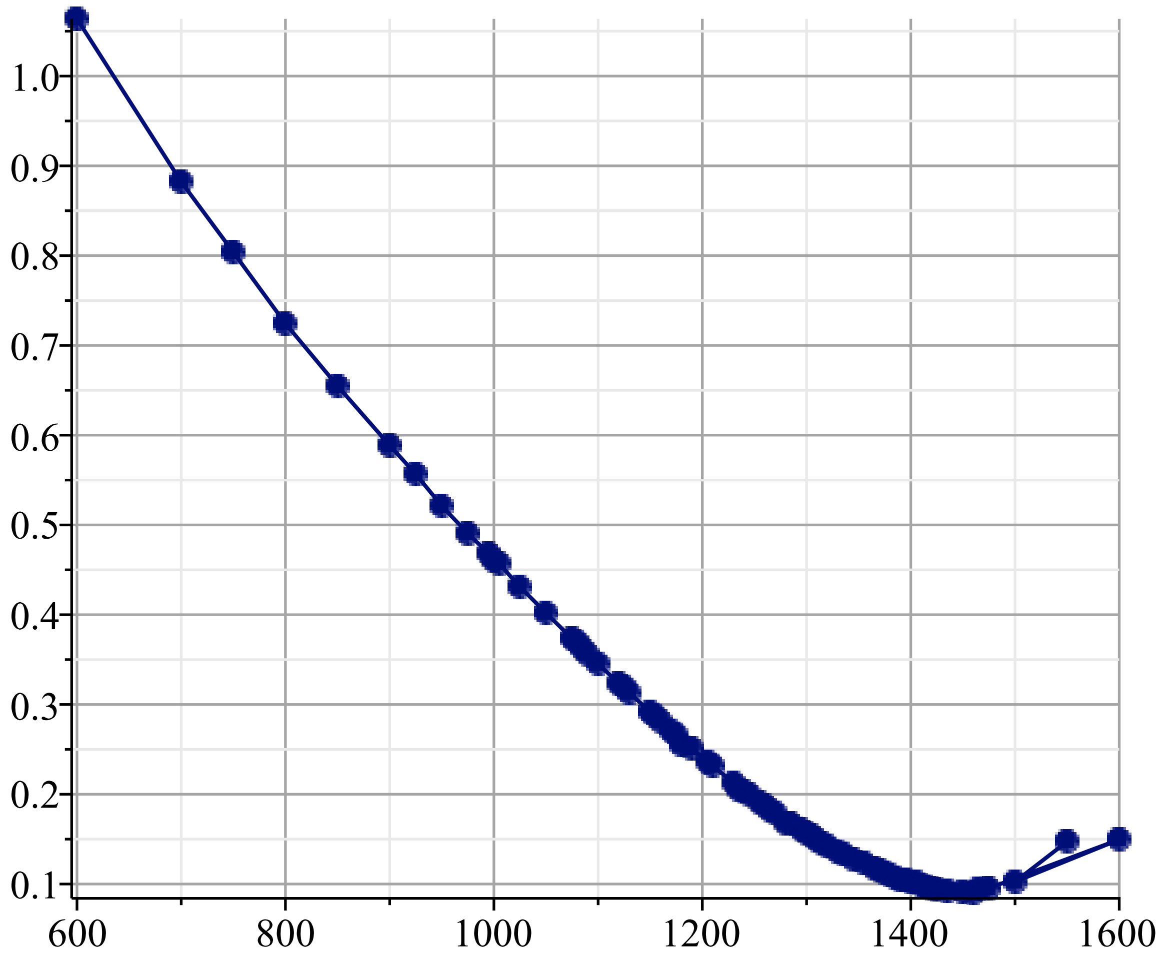
[79, *December*, 2006, 1550, "SXMLJ", 0.20000, 0, 0.05000, 0.50000, ...],
[80, *December*, 2006, 1500, "SXMLT", 0.25000, 0, 0.10000, 0.30000, ...],
[81, *December*, 2006, 1600, "SPBLT", 0.05000, -0.15000, 0.15000, 0.10000, ...],
[82, *December*, 2006, 1475, "SXZLO", 0.50000, -0.05000, 0.25000, 0.65000, ...],
[83, *December*, 2006, 1465, "SXZLM", 0.85000, 0.25000, 0.50000, 1, ...],
[84, *December*, 2006, 1460, "SXZLL", 1.10000, -0.25000, 0.75000, 1, ...],
[85, *December*, 2006, 1450, "SXZLJ", 1.65000, 0, 1.20000, 1.70000, ...],
[86, *December*, 2006, 1435, "SXZLG", 2.90000, -0.20000, 2.60000, 3.40000, ...],
[..., ..., ..., ..., ..., ..., ..., ..., ..., ...]]

Compute the implied volatility for these dates and plot the results:

> $L1 := Matrix([seq([Dec2006[i, "Strike"], ImpliedVolatility(Dec2006[i, "Ask"], S0, Dec2006[i, "Strike"], Time[1], r, d)], i = 1 .. NumRows(Dec2006))], datatype = float[8])$

$$L1 := \begin{bmatrix} \textit{72 x 2 Matrix} \\ \textit{Data Type: } float_8 \\ \textit{Storage: rectangular} \\ \textit{Order: Fortran\_order} \end{bmatrix} \quad (1.10)$$

> $dataplot(L1[.., 1], L1[.., 2], gridlines, size = [600, "golden"])$



Implied volatilities for options maturing in December 2007.

> $Time_2 := YearFraction(\text{"October 27, 2006"}, NthWeekday(3, Friday, December, 2007))$

$$Time_2 := 1.15068 \quad (1.11)$$

Extract a subset of the DataFrame corresponding to the observations in December 2007:

> $Dec2007 := DF[DF[\text{"Month"}] =\sim December \textbf{ and } DF[\text{"Year"}] =\sim 2007]$

Dec2007 := [[, "Month", "Year", "Strike", "Symbol", "Last", "Change", "Bid", "Ask", ...], **(1.12)**
[256, *December*, 2007, 1650, "SZVLK", 7.20000, 0.20000, 5.70000, 7.30000, ...],
[257, *December*, 2007, 1600, "SZVLO", 14, -0.10000, 13.10000, 15.10000, ...],
[258, *December*, 2007, 1525, "SZVLE", 15.50000, 0.50000, 31.80000, 34.80000, ...],
[259, *December*, 2007, 1500, "SZVLT", 42.20000, -0.60000, 40.60000, 43.60000, ...
],
[260, *December*, 2007, 1450, "SZTLS", 66.50000, 6.10000, 63.10000, 66.10000, ...],
[261, *December*, 2007, 1425, "SZTLE", 66.70000, 8.90000, 76.30000, 79.30000, ...],
[262, *December*, 2007, 1375, "SZTLO", 107, 4, 106.20000, 109.20000, ...],

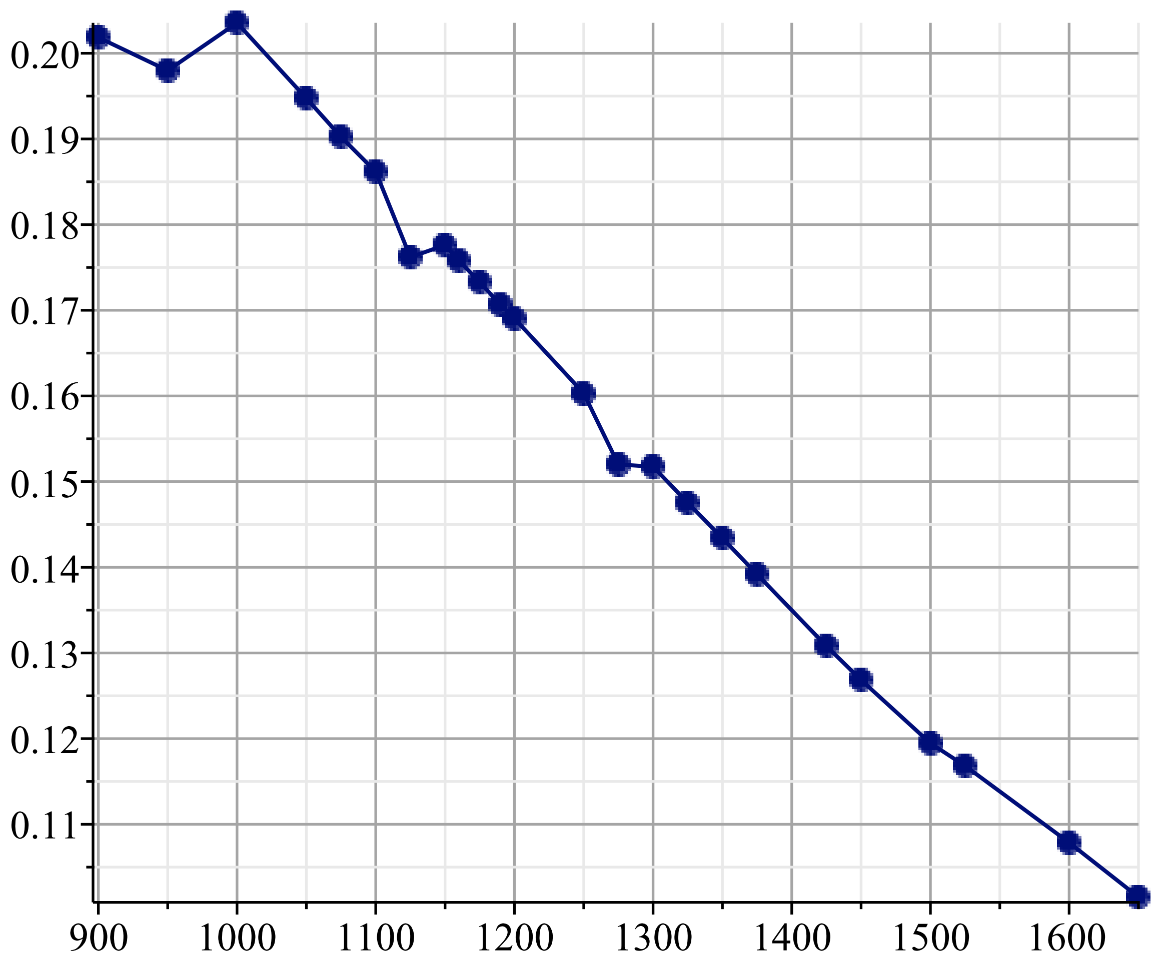
[263, *December*, 2007, 1350, "SZTLK", 124.50000, 2, 122.60000, 125.60000, ...],
[..., ..., ..., ..., ..., ..., ..., ..., ..., ...]]

Compute the implied volatility for these dates and plot the results:

```
> L2 := Matrix([seq( [ Dec2007[i, "Strike"], ImpliedVolatility( Dec2007[i, "Ask"], S0,
    Dec2007[i, "Strike"], Time[2], r, d ) ], i = 1 .. NumRows(Dec2007) ) ], datatype
    = float[8])
```

$$L2 := \begin{bmatrix} \textit{24 x 2 Matrix} \\ \textit{Data Type: } float_8 \\ \textit{Storage: rectangular} \\ \textit{Order: Fortran\_order} \end{bmatrix} \quad \textbf{(1.13)}$$

```
> dataplot(L2[ .., 1], L2[ .., 2], gridlines, size = [600, "golden"])
```



We will use the following model for the volatility surface.

> $\Sigma := (S, T, K) \rightarrow \alpha_1 + \alpha_2 \ln\left(\frac{K}{S}\right) + \frac{\alpha_3}{\sqrt{T}} + \alpha_4 \ln\left(\frac{K}{S}\right)^2 + \frac{\alpha_5 \ln\left(\frac{K}{S}\right)}{\sqrt{T}} + \frac{\alpha_6 \ln\left(\frac{K}{S}\right)^2}{\sqrt{T}}$

$$\Sigma := (S, T, K) \rightarrow \alpha_1 + \alpha_2 \ln\left(\frac{K}{S}\right) + \frac{\alpha_3}{\sqrt{T}} + \alpha_4 \ln\left(\frac{K}{S}\right)^2 + \frac{\alpha_5 \ln\left(\frac{K}{S}\right)}{\sqrt{T}} + \frac{\alpha_6 \ln\left(\frac{K}{S}\right)^2}{\sqrt{T}} \quad \textbf{(1.14)}$$

We can compute the corresponding Black-Scholes price as a function of strike and maturity.

> $C := BlackScholesPrice\left(S0, K, T, \Sigma(S0, T, K), r, d\right)$

$$\begin{aligned} C := {} & 1377.00000\, \mathrm{e}^{-0.01820\, T} \Bigg(0.50000 + 0.50000\, \mathrm{erf}\Bigg(\Bigg(0.70711\, \Bigg(\ln\left(\frac{1377.00000}{K}\right) \\ & + 0.03500\, T + 0.50000\, \Bigg(\alpha_1 + \alpha_2 \ln(0.00073\, K) + \frac{\alpha_3}{\sqrt{T}} + \alpha_4 \ln(0.00073\, K)^2 \\ & + \frac{\alpha_5 \ln(0.00073\, K)}{\sqrt{T}} + \frac{\alpha_6 \ln(0.00073\, K)^2}{\sqrt{T}}\Bigg)^2 T\Bigg)\Bigg) \Bigg/ \Bigg(\Bigg(\alpha_1 + \alpha_2 \ln(0.00073\, K) \\ & + \frac{\alpha_3}{\sqrt{T}} + \alpha_4 \ln(0.00073\, K)^2 + \frac{\alpha_5 \ln(0.00073\, K)}{\sqrt{T}} + \frac{\alpha_6 \ln(0.00073\, K)^2}{\sqrt{T}}\Bigg) \sqrt{T}\Bigg)\Bigg)\Bigg) \\ & - 1.00000\, \mathrm{e}^{-0.05320\, T} K \Bigg(0.50000 + 0.50000\, \mathrm{erf}\Bigg(\Bigg(0.70711\, \Bigg(\ln\left(\frac{1377.00000}{K}\right) \\ & + 0.03500\, T + 0.50000\, \Bigg(\alpha_1 + \alpha_2 \ln(0.00073\, K) + \frac{\alpha_3}{\sqrt{T}} + \alpha_4 \ln(0.00073\, K)^2 \\ & + \frac{\alpha_5 \ln(0.00073\, K)}{\sqrt{T}} + \frac{\alpha_6 \ln(0.00073\, K)^2}{\sqrt{T}}\Bigg)^2 T\Bigg)\Bigg) \Bigg/ \Bigg(\Bigg(\alpha_1 + \alpha_2 \ln(0.00073\, K) \\ & + \frac{\alpha_3}{\sqrt{T}} + \alpha_4 \ln(0.00073\, K)^2 + \frac{\alpha_5 \ln(0.00073\, K)}{\sqrt{T}} + \frac{\alpha_6 \ln(0.00073\, K)^2}{\sqrt{T}}\Bigg) \sqrt{T}\Bigg) \\ & - 0.70711\, \Bigg(\alpha_1 + \alpha_2 \ln(0.00073\, K) + \frac{\alpha_3}{\sqrt{T}} + \alpha_4 \ln(0.00073\, K)^2 \\ & + \frac{\alpha_5 \ln(0.00073\, K)}{\sqrt{T}} + \frac{\alpha_6 \ln(0.00073\, K)^2}{\sqrt{T}}\Bigg) \sqrt{T}\Bigg)\Bigg) \end{aligned} \quad \textbf{(1.15)}$$

We can use non-linear fitting routines from the statistics data to find the values of $\alpha_1, \ldots, \alpha_6$ that best fit our data. Construct a matrix of parameters and a vector of the corresponding value of the objective function.

> $B := \langle\langle Times\rangle | convert(DF["\text{Strike}"], Matrix)\rangle$

$$B := \begin{bmatrix} \textit{308 x 2 Matrix} \\ \textit{Data Type: anything} \\ \textit{Storage: rectangular} \\ \textit{Order: Fortran\_order} \end{bmatrix} \quad \textbf{(1.16)}$$

> $V := convert(DF["\text{Ask}"], Vector[column])$

$$V := \begin{bmatrix} \textit{1 .. 308 } Vector_{column} \\ \textit{Data Type: anything} \\ \textit{Storage: rectangular} \\ \textit{Order: Fortran\_order} \end{bmatrix} \quad \textbf{(1.17)}$$

> $\beta 1 := NonlinearFit\left(C, B, V, [T, K], parameternames = [\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6], output = parametervector\right)$

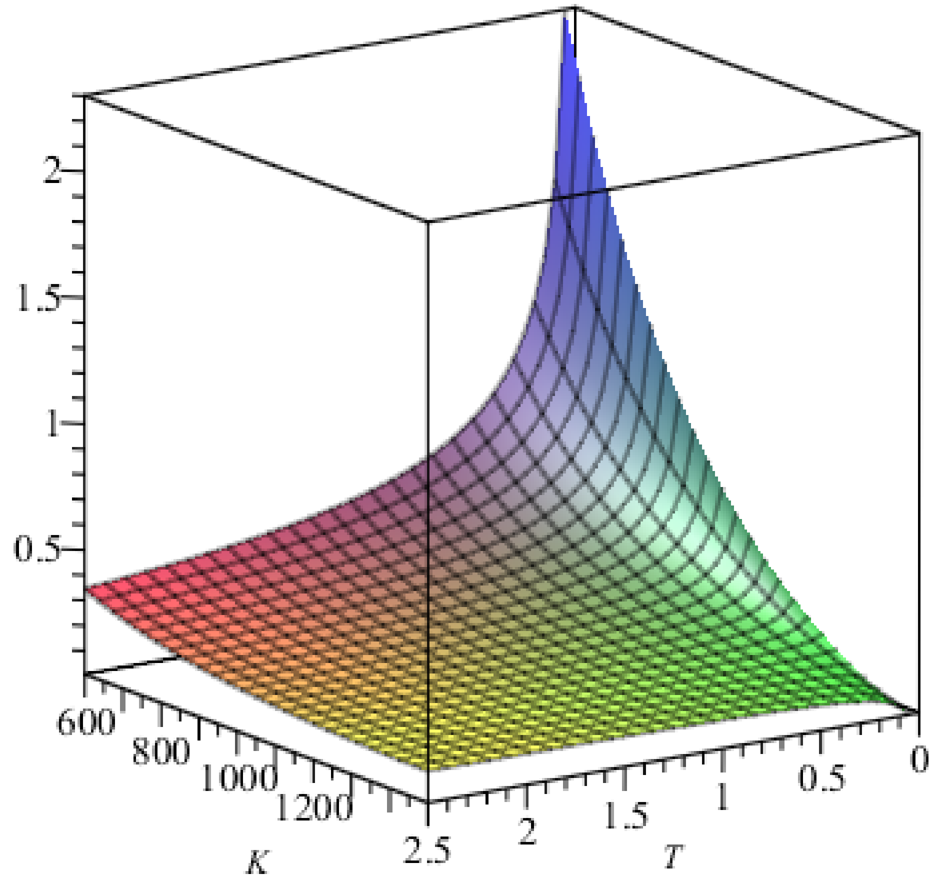
$$\beta 1 := \begin{bmatrix} 0.14393 \\ 0.12599 \\ -0.01069 \\ -0.03301 \\ -0.30394 \\ 0.40480 \end{bmatrix} \quad \textbf{(1.18)}$$

Here is the corresponding implied volatility function.

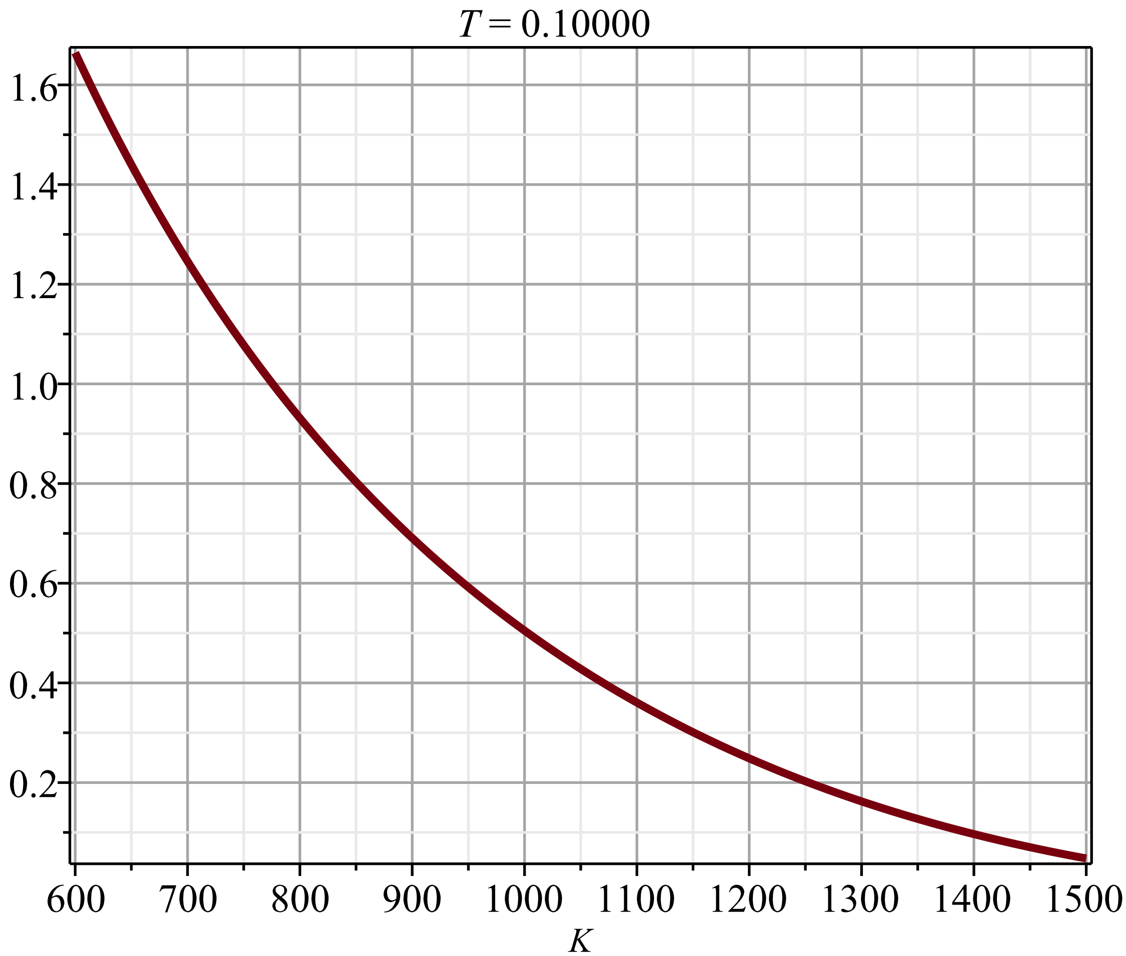
> $F1 := \Sigma(S0, T, K) \Big|_{\alpha = \beta 1}$

$$F1 := 0.14393 + 0.12599 \ln\left(\frac{1}{1377} K\right) - \frac{0.01069}{\sqrt{T}} - 0.03301 \ln\left(\frac{1}{1377} K\right)^2 - \frac{0.30394 \ln\left(\frac{1}{1377} K\right)}{\sqrt{T}} + \frac{0.40480 \ln\left(\frac{1}{1377} K\right)^2}{\sqrt{T}} \quad \textbf{(1.19)}$$

> $plot3d(F1, T = 0 .. 2.5, K = 600 .. 1500)$



> *animate*(*plot*, [*F1*, *K* = 600 ..1500, *thickness* = 3], *T* = 0.1 ..3, *axes* = *boxed*, *gridlines*, *size* = [600, "golden"])



Here is another way to estimate these parameters.

> $U := Vector(1..M)$

$$U := \left[\begin{array}{c} \text{1 .. 308 } Vector_{column} \\ \text{Data Type: anything} \\ \text{Storage: rectangular} \\ \text{Order: Fortran\_order} \end{array}\right] \quad (1.20)$$

> **for** i **to** M **do**
$U_i := ImpliedVolatility\left(V_i, S0, B_{i,2}, B_{i,1}, r, d\right)$
end do:

> $\beta 2 := NonlinearFit\left(\Sigma(S0, T, K), B, U, [T, K], parameternames = \left[\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6\right], output = parametervector\right)$

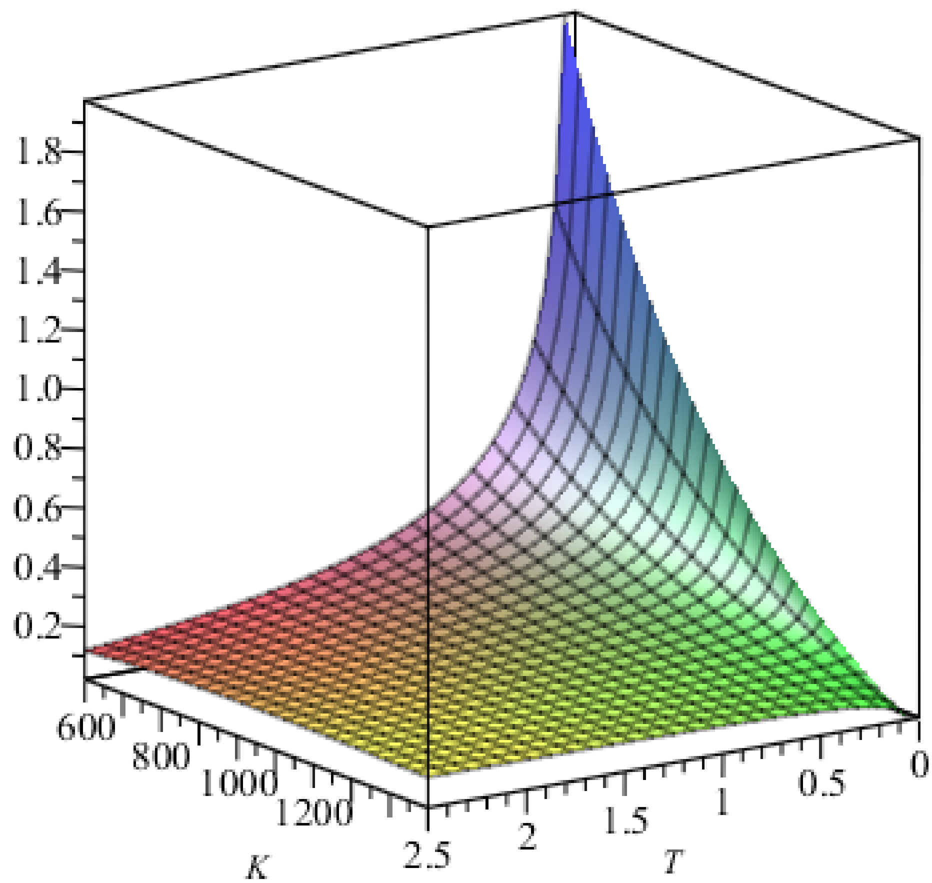
$$\beta 2 := \begin{bmatrix} 0.13011 \\ 0.17037 \\ 0.00115 \\ -0.26597 \\ -0.36133 \\ 0.28070 \end{bmatrix} \tag{1.21}$$

Here is the corresponding implied volatility function.

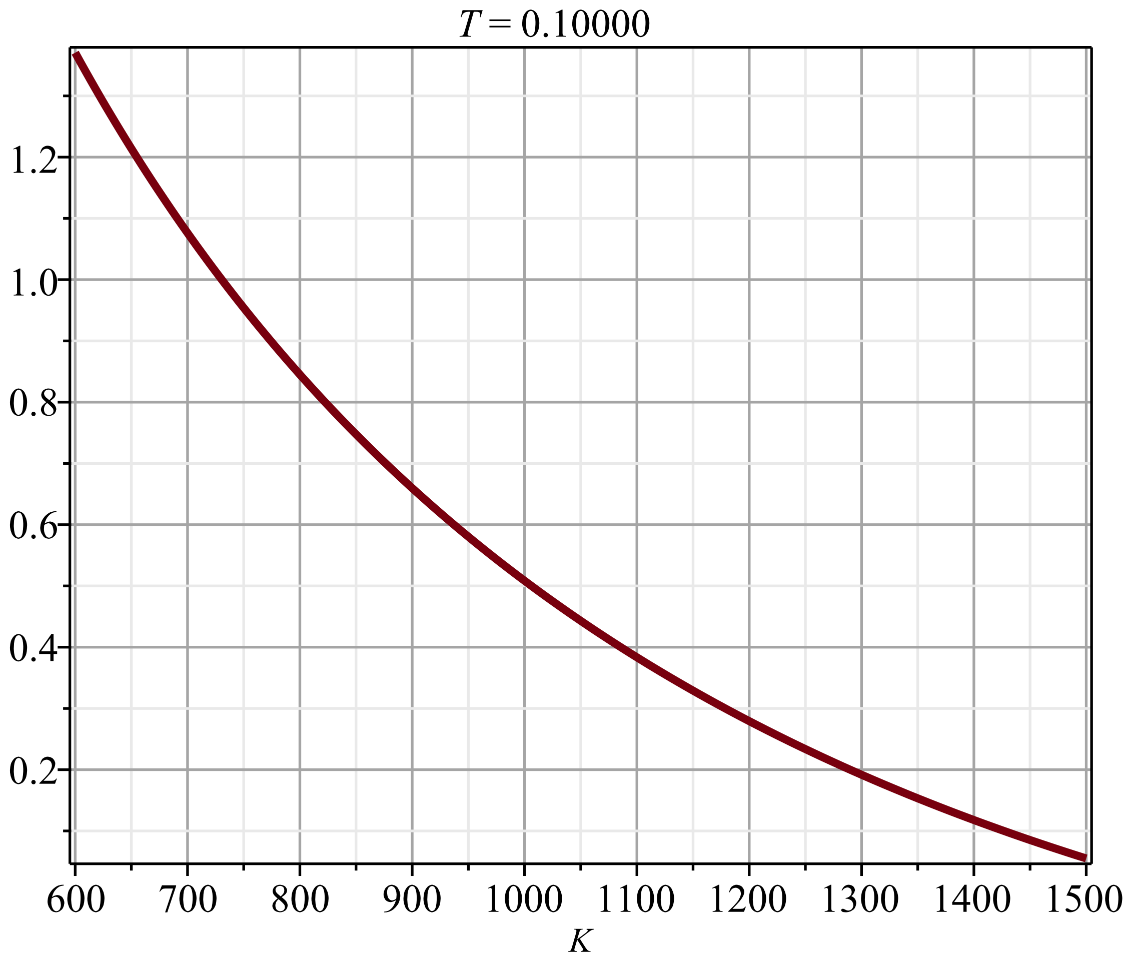
> $F2 := \Sigma(S0, T, K)\Big|_{\alpha = \beta 2}$

$$F2 := 0.13011 + 0.17037 \ln\left(\frac{1}{1377} K\right) + \frac{0.00115}{\sqrt{T}} - 0.26597 \ln\left(\frac{1}{1377} K\right)^2 - \frac{0.36133 \ln\left(\frac{1}{1377} K\right)}{\sqrt{T}} + \frac{0.28070 \ln\left(\frac{1}{1377} K\right)^2}{\sqrt{T}} \tag{1.22}$$

> $plot3d(F2, T = 0 ..2.5, K = 600 ..1500)$



> *animate*(*plot*, [*F2*, *K* = 600 ..1500, *thickness* = 3], *T* = 0.1 ..3, *gridlines*, *axes* = *boxed*, *size* = [600, "golden"])



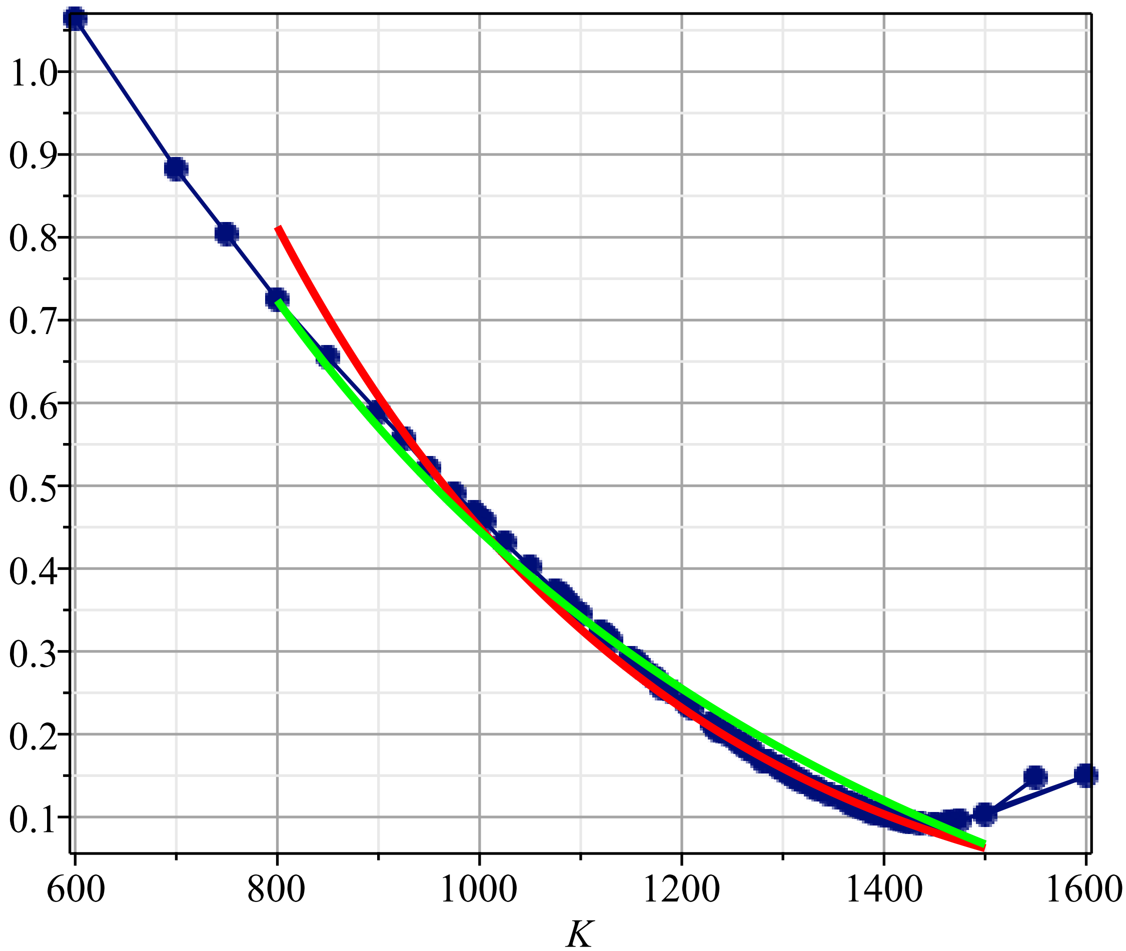
We can compare both fits with the actual implied volatilities.

> $P := dataplot(L1[\,..,1], L1[\,..,2])$:

> $P1 := plot\left(F1\Big|_{T=Time_1}, K = 800\,..1500, thickness = 3, color = red\right)$:

> $P2 := plot\left(F2\Big|_{T=Time_1}, K = 800\,..1500, thickness = 3, color = green\right)$:

> $display(P, P1, P2, axes = boxed, gridlines, size = [600, \text{"golden"}])$

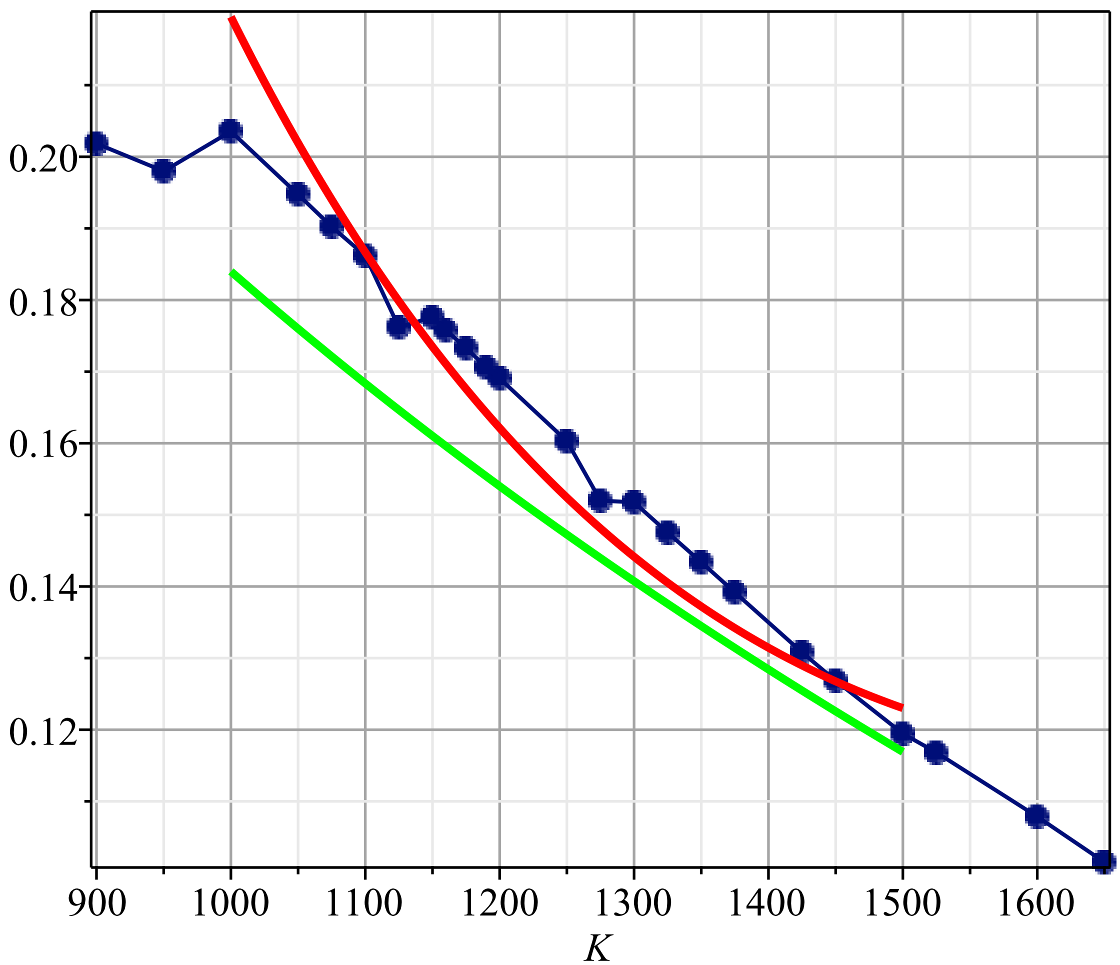


> $P := dataplot(L2[..,1], L2[..,2])$:

> $P1 := plot\left(F1 \Big|_{T = Time_2}, K = 1000..1500, thickness = 3, color = red\right)$:

> $P2 := plot\left(F2 \Big|_{T = Time_2}, K = 1000..1500, thickness = 3, color = green\right)$:

> $display(P, P1, P2, gridlines, axes = boxed, size = [600, \text{"golden"}])$



Modeling with Local Volatility

We will consider the same model for the local volatility except that in this case we will use parameters that were fit to some market data.

> *restart* :
> *with*(*Finance*) :

> $\Sigma := \alpha_1 + \alpha_2 \ln\left(\frac{K}{S0}\right) + \frac{\alpha_3 \cdot 1}{\sqrt{T}} + \alpha_4 \ln\left(\frac{K}{S0}\right)^2 + \frac{\alpha_5 \ln\left(\frac{K}{S0}\right)}{\sqrt{T}} + \frac{\alpha_6 \ln\left(\frac{K}{S0}\right)^2}{\sqrt{T}}$

$$\Sigma := \alpha_1 + \alpha_2 \ln\left(\frac{K}{S0}\right) + \frac{\alpha_3}{\sqrt{T}} + \alpha_4 \ln\left(\frac{K}{S0}\right)^2 + \frac{\alpha_5 \ln\left(\frac{K}{S0}\right)}{\sqrt{T}} + \frac{\alpha_6 \ln\left(\frac{K}{S0}\right)^2}{\sqrt{T}} \qquad \textbf{(2.1)}$$

> $\alpha_1 := 0.1581 : \alpha_2 := -0.2777 : \alpha_3 := -0.0050 : \alpha_4 := -0.5011 : \alpha_5 := 0.1103 : \alpha_6 := 0.5909 :$

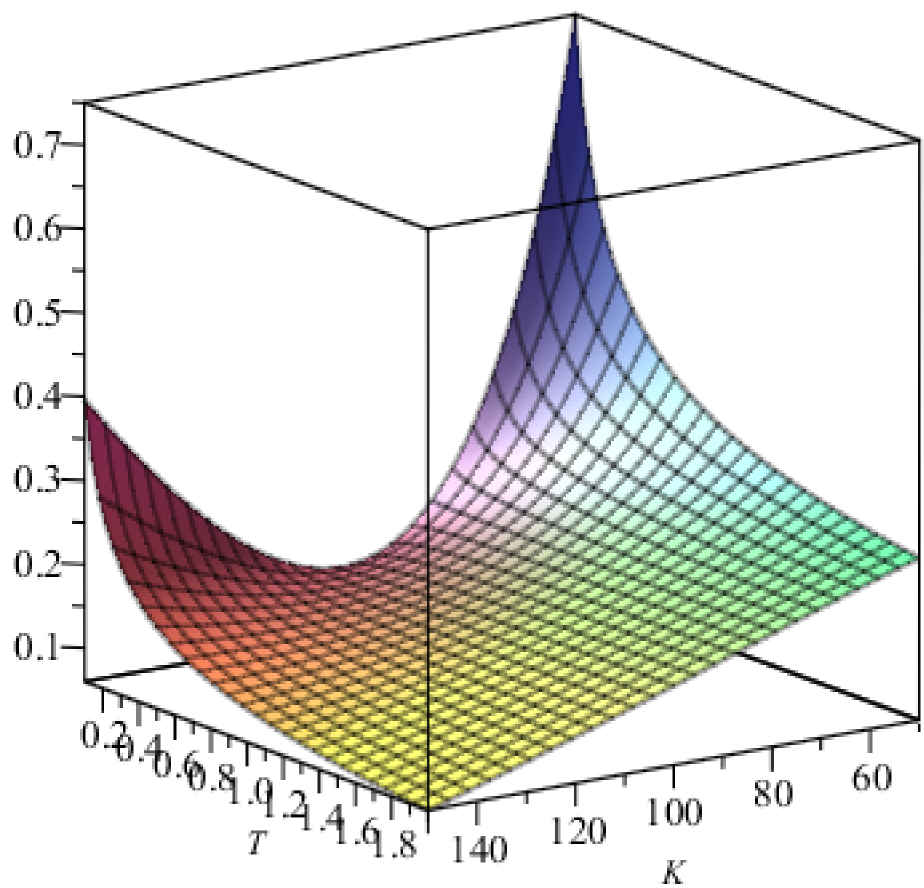
> $S0 := 100$

$$S0 := 100 \tag{2.2}$$

> Σ

$$0.15810 - 0.27770 \ln\left(\frac{1}{100} K\right) - \frac{0.00500}{\sqrt{T}} - 0.50110 \ln\left(\frac{1}{100} K\right)^2 + \frac{0.11030 \ln\left(\frac{1}{100} K\right)}{\sqrt{T}} + \frac{0.59090 \ln\left(\frac{1}{100} K\right)^2}{\sqrt{T}} \tag{2.3}$$

> $plot3d(\Sigma, K = 50 ..150, T = 0.1 ..2)$



Consider two functions. The first one returns the Black-Scholes price of a European call option for our model. The second one returns the Black-Scholes price of a European put option for our model. We will assume that these functions are given two us (e.g. obtained by interpolating the market data) and will try to determine the corresponding local volatility term structure.

> $C := BlackScholesPrice(S0, K, T, \Sigma, 0.05, 0.01, 'call') :$

> $P := BlackScholesPrice(S0, K, T, \Sigma, 0.05, 0.01, 'put') :$

Construct the corresponding local volatility surface.

> $V := LocalVolatilitySurface(0.05, 0.01, C, K, T) :$

> $S1 := \langle seq(i, i = 50 .. 200) \rangle$

$$S1 := \begin{bmatrix} \text{1 .. 151 } Vector_{column} \\ \text{Data Type: anything} \\ \text{Storage: rectangular} \\ \text{Order: Fortran\_order} \end{bmatrix} \qquad (2.4)$$

> $T1 := \langle seq(0.02\, i, i = 1 .. 50) \rangle$

$$T1 := \begin{bmatrix} \text{1 .. 50 } Vector_{column} \\ \text{Data Type: anything} \\ \text{Storage: rectangular} \\ \text{Order: Fortran\_order} \end{bmatrix} \qquad (2.5)$$

> $V := LocalVolatility(C, S1, T1, 0.05, 0.01, T, K)$

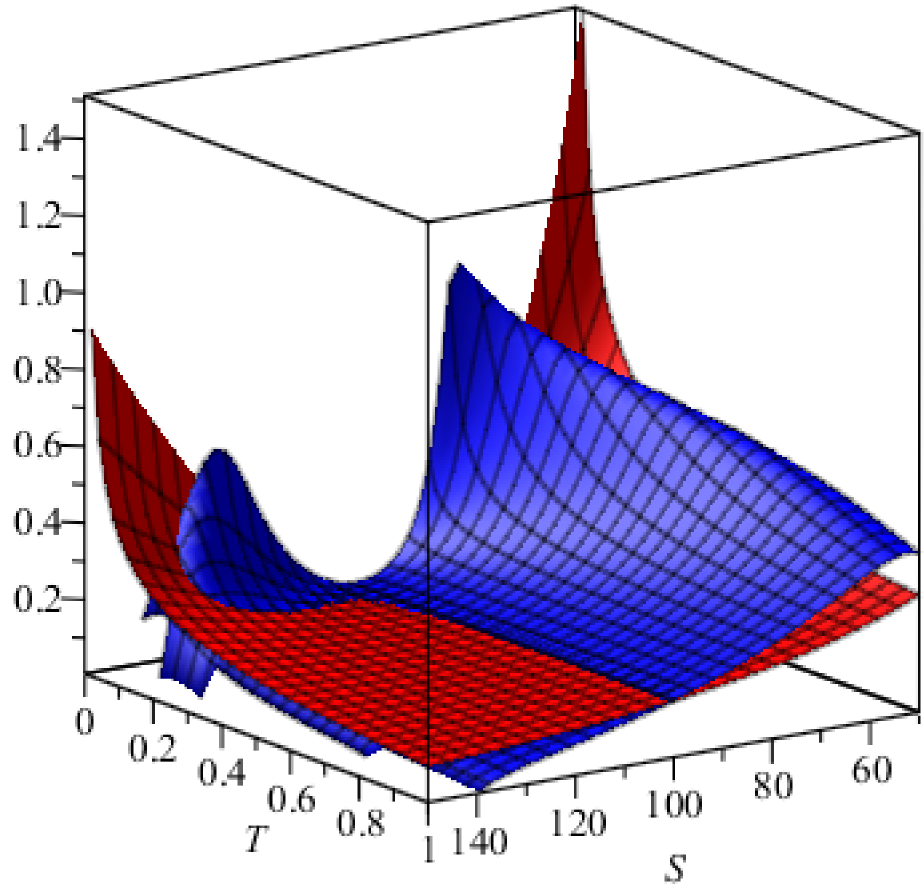
$$V := \begin{bmatrix} \text{151 x 50 Matrix} \\ \text{Data Type: anything} \\ \text{Storage: rectangular} \\ \text{Order: Fortran\_order} \end{bmatrix} \qquad (2.6)$$

> $LV := LocalVolatility(C, S, T, 0.05, 0.01, T, K) :$

> $P1 := plot3d(LV, S = 50 .. 150, T = 0.05 .. 1, color = blue) :$

> $P2 := plot3d(\Sigma, K = 50 .. 150, T = 0 .. 1, color = red) :$

> $plots_{display}(P1, P2)$



We can construct the corresponding local volatility surface and implied volatility surface.

> $LVS := LocalVolatilitySurface(T1, S1, V)$:

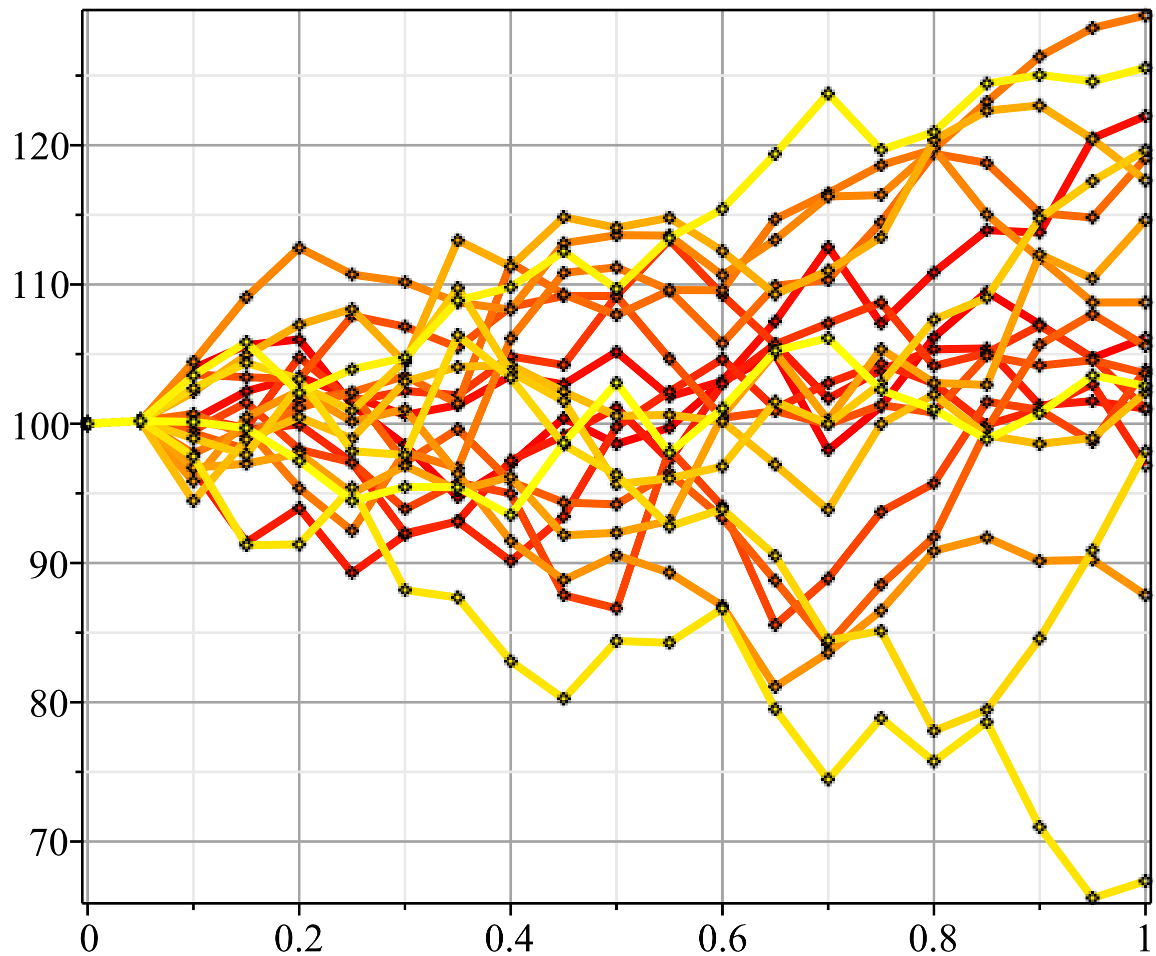
> $IVS := ImpliedVolatilitySurface(\Sigma, T, K)$:

We can now construct a Black-Scholes process which has the volatility structure we just obtained.

> $X := BlackScholesProcess(100, LVS, 0.05, 0.01)$

$$X := \_X \quad (2.7)$$

> $PathPlot(X(t), t = 0..1, timesteps = 20, replications = 20, thickness = 3, color = red..yellow, axes = boxed, gridlines, size = [600, "golden"])$



> $P1 := subs\left(K = S0\, x, \frac{1\, P}{S0}\right)$:

> $C1 := subs\left(K = S0\, x, \frac{1\, C}{S0}\right)$:

> $T := 'T'$

$$T := T \quad \textbf{(2.8)}$$

As an alternative, we can use the implied volatility surface to construct an implied trinomial tree.

> $TT := ImpliedTrinomialTree(100, 0.05, 0.01, IVS, 1, 100)$:

> $GetLocalVolatility(TT, 99, 99, 0.05)$

$$0.1531851114 \quad \textbf{(2.9)}$$

> $IVS(1.0, 100)$

$$0.1531000000 \quad \textbf{(2.10)}$$

Compute some option prices. We can use Monte-Carlo simulation to price European-style options and lattice methods to price American-style options.

> $ExpectedValue\left(\max(X(1)-100, 0), timesteps = 100, replications = 10^4\right)$

$$[value = 8.561588770, standarderror = 0.09544293863] \quad (2.11)$$

> $E := EuropeanOption(t \rightarrow \max(t-100, 0), 1.0)$

$$E := \textbf{module}(\ )\ \ldots\ \textbf{end module} \quad (2.12)$$

> $LatticePrice(E, TT, 0.0)$

$$8.477754630 \quad (2.13)$$

Here is an example of an Asian-type option with European exercise.

> $ExpectedValue\left(\max\left(\int_0^1 X(u)\,du - 100, 0\right), timesteps = 100, replications = 10^3\right)$

$$[value = 4.344002588, standarderror = 0.1776650336] \quad (2.14)$$