KU LEUVEN

LRISK

RISK CONCILE

Boosting Derivatives Pricing with Machine Learning

Jan De Spiegeleer

Risk Analytics Symposium
2019
Chicago

Joint work with Dilip Madan, Wim Schoutens, Sofie Reyners (KU LEUVEN)

Content

- Machine Learning for Quant Problems
- Gauss Process Regression Explained
- Pricing of Exotics
- Comparing GPR with GBM
- Conclusions

Introduction

Derivative pricing and risk management is time-consuming

- Exotic option pricing

→ *Monte Carlo simulations*

- VaR and ES

→ *Scenarios*

- Counter-party exposure

… and markets are moving !

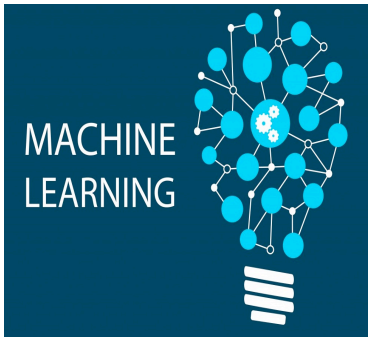
time-consuming algorithms

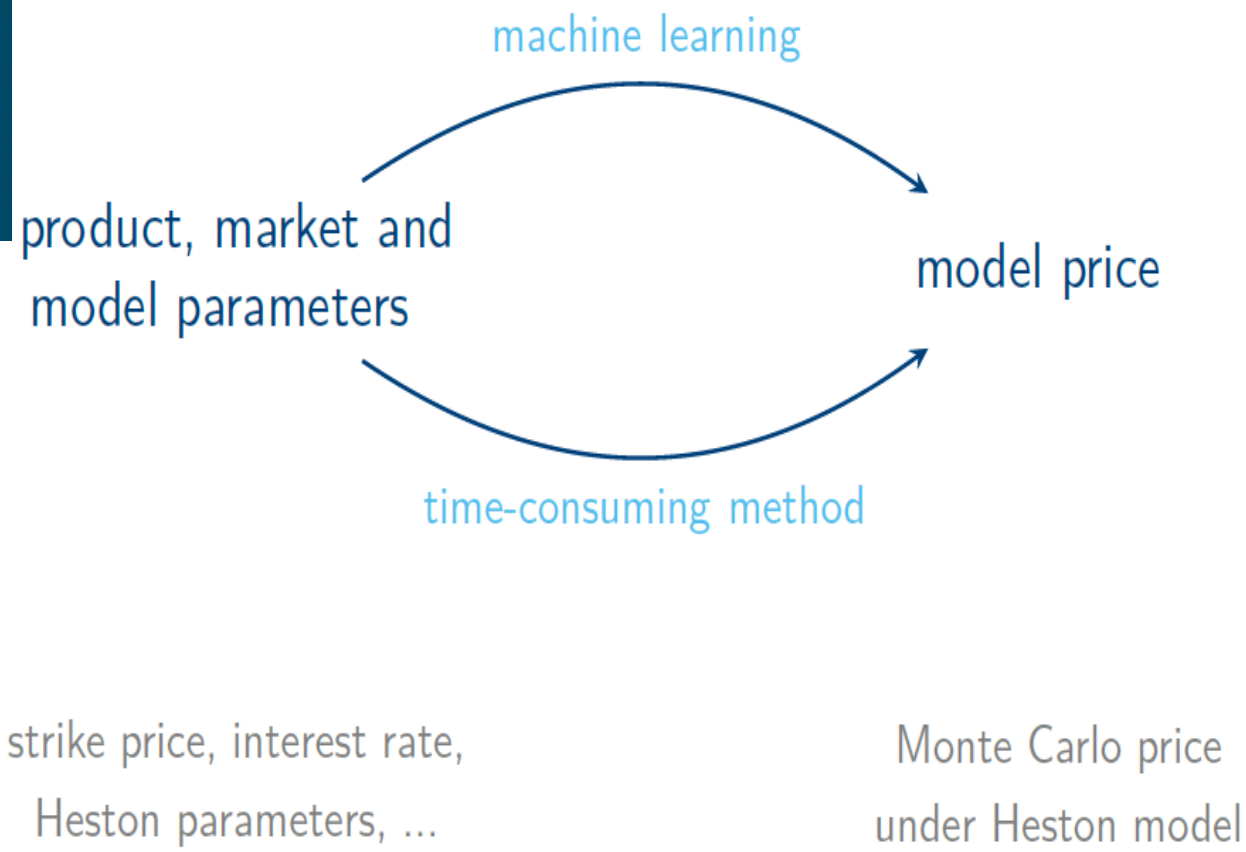
$\updownarrow$

continuously moving markets

$\rightarrow$ prices are outdated when available, overnight calculations cannot be performed in one night, …

Machine Learning techniques can help here

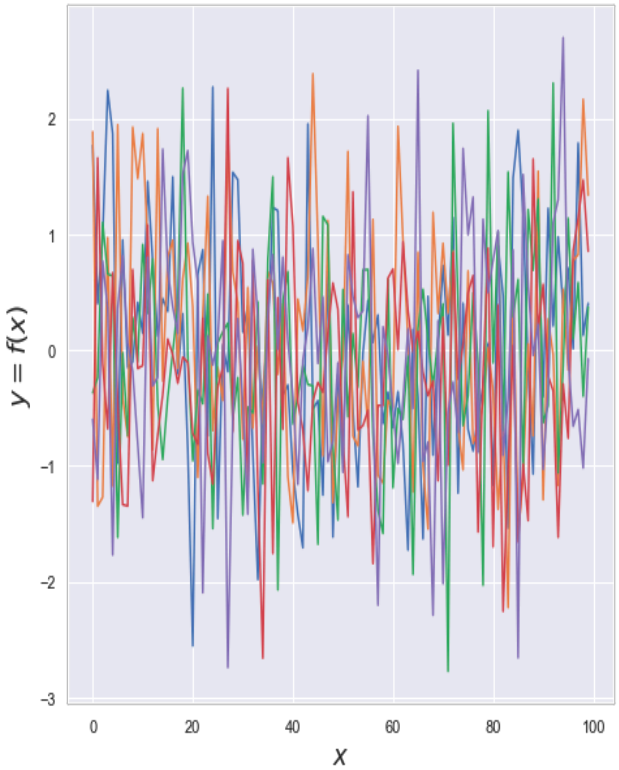




When training is completed, prediction is extremely fast!

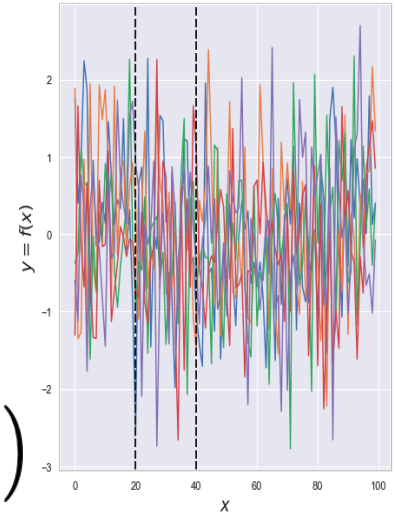
Gaussian Process Regression

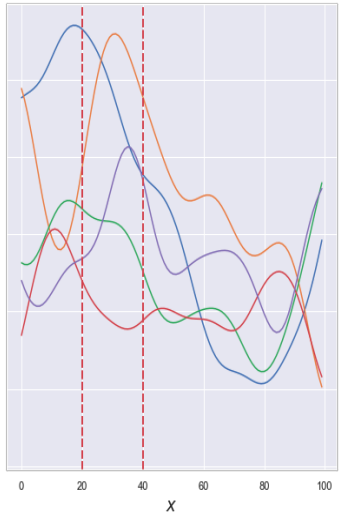
Assume f(x) to be multi-variate Normal (GP)



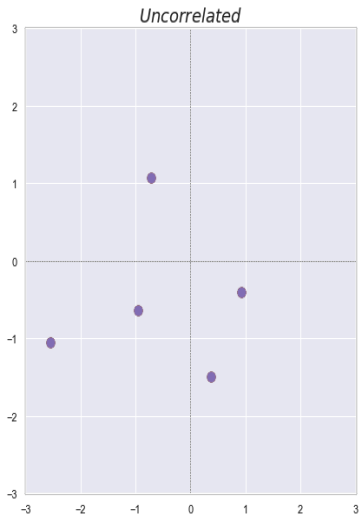
“Forcing” a Structure upon the GP

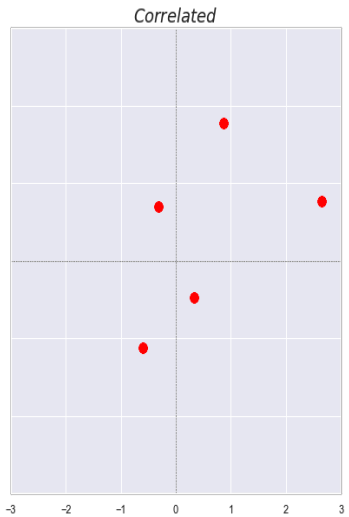
$$f \sim N(0, I)$$

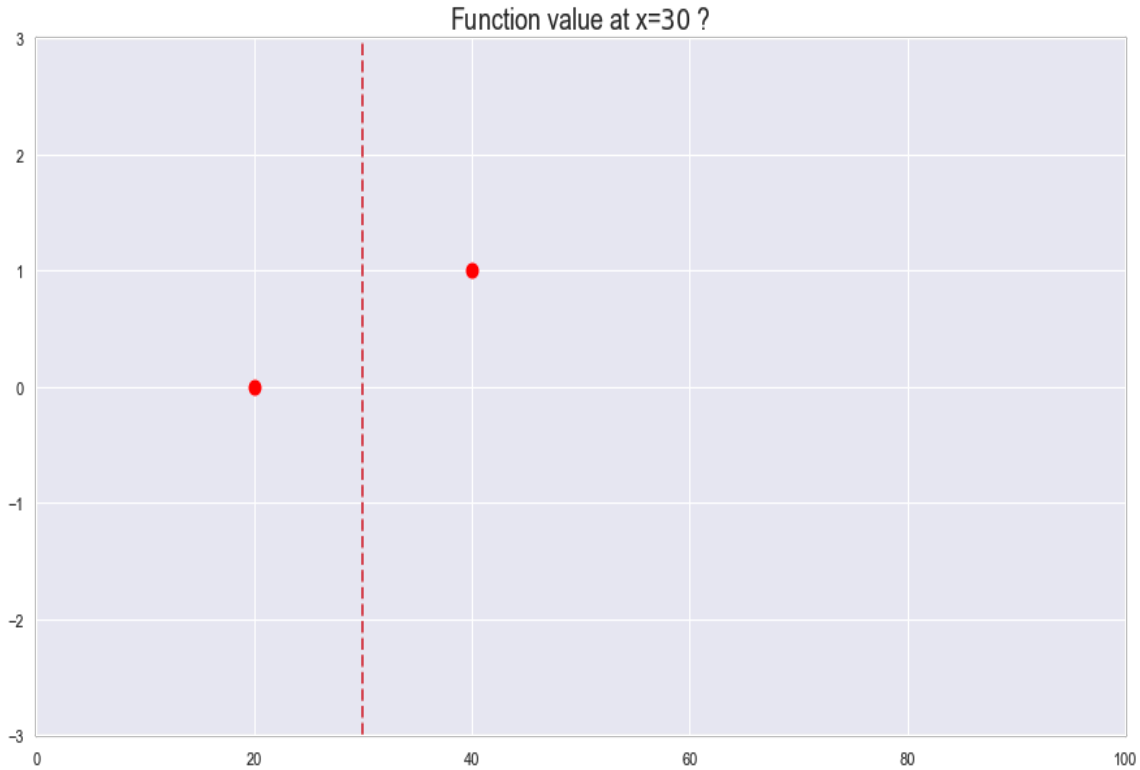




$$f \sim N(0, \Sigma)$$







We have the function value 2 points $x_1 = 20$ and $x_2 = 40$.
The corresponding values are $f_1 = 0$ and $f_2 = 1$.
What is the value in point x_3 ?

The only thing we have are the data-points (f_1 and f_2) and the fact that f is a Gaussian process

The only freedom we have is the choice of the Kernel function $K()$ and its hyper-parameters

Using Kernels

The kernel $K(a,b)$ of two points a and b gives a measure of the "distance" of these points

Examples of kernel functions

- **Radial Basis Function** $K(a,b) = \exp(\frac{1}{2l^2} \mid\mid a - b \mid\mid^2))$
- **Matern** $K(a,b) = \sigma^2 \frac{1}{\Gamma(\nu)2^{\nu-1}} \left(\gamma\sqrt{2\nu}d(a/l, b/l)\right)^{\nu} K_\nu \left(\gamma\sqrt{2\nu}d(a/l, b/l)\right)$
- **RationalQuadratic** $K(a,b) = \left(1 + \frac{d(a,b)^2}{2\alpha l^2}\right)^{-\alpha}$
- **ExpSineSquared** $K(a,b) = \exp\left(-2\left(\sin(\pi/p * d(a,b))/l\right)^2\right)$

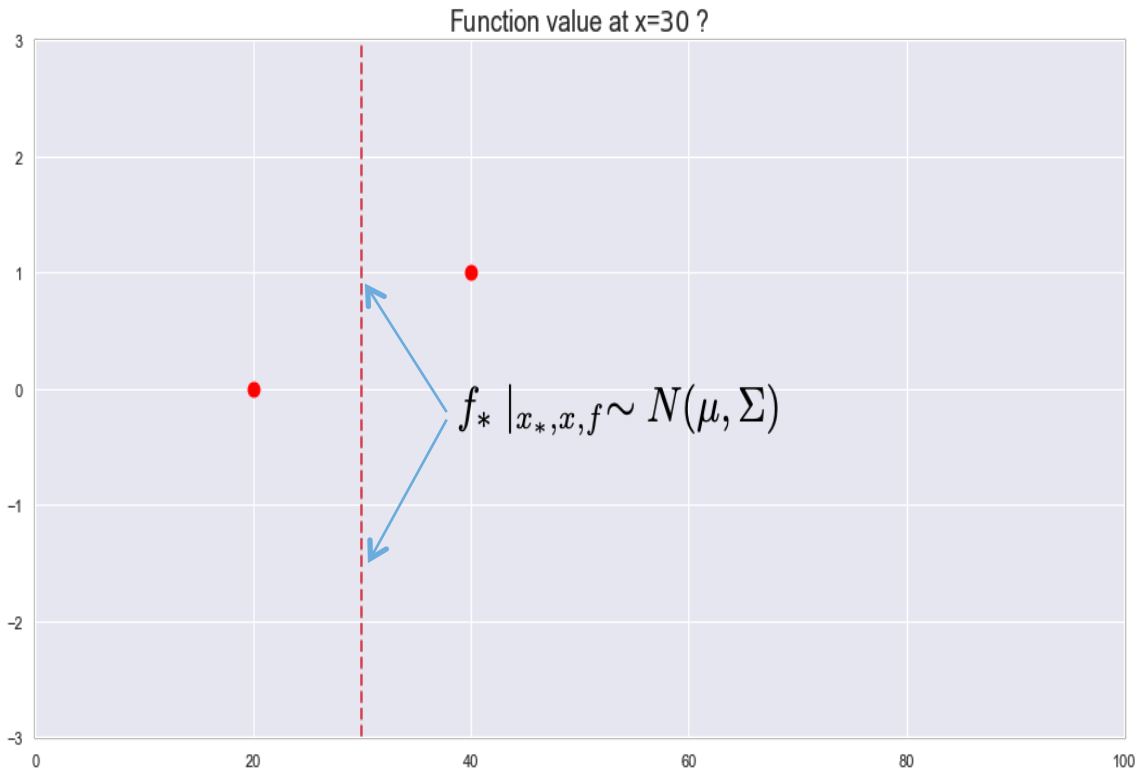
Reference:
Carl Eduard Rasmussen and Christopher K.I. Williams, "Gaussian Processes for Machine Learning", MIT Press 2006,

Model = Data

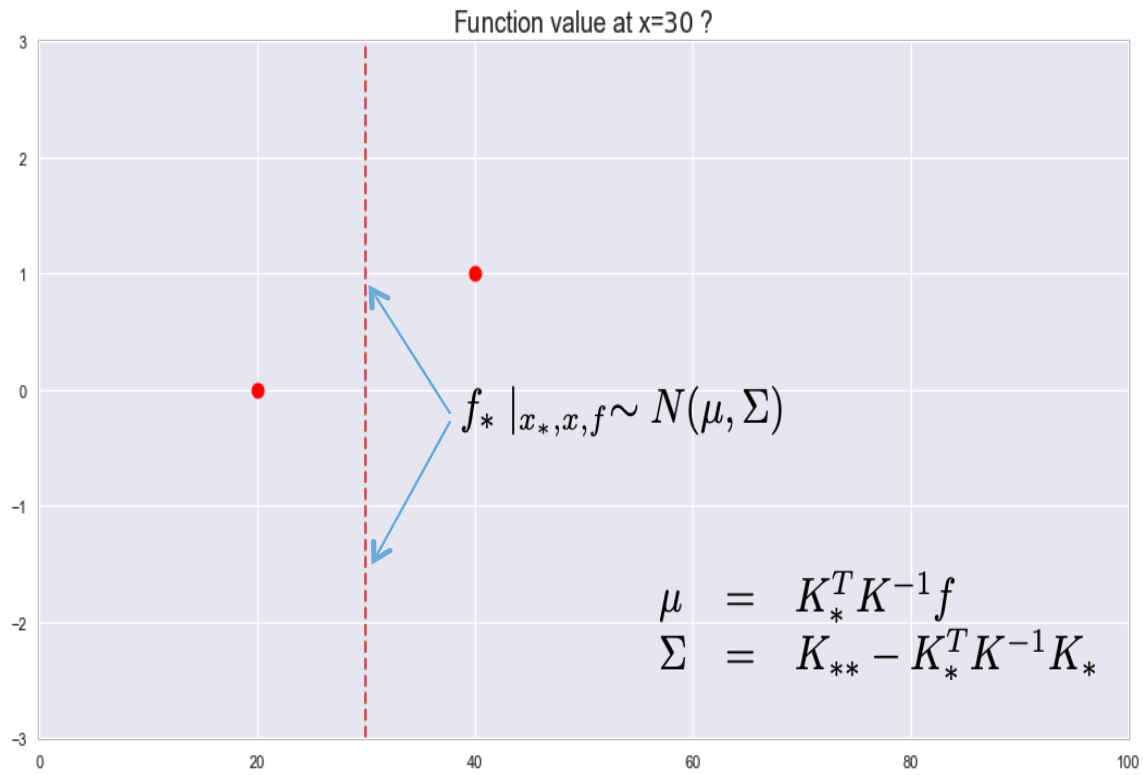
$$\begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{bmatrix} \right)$$

$$\begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{bmatrix}\right)$$

$$\begin{bmatrix} f \\ f_* \end{bmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} K & K_* \\ K_*^T & K_{**} \end{bmatrix}\right)$$



$$\begin{bmatrix} f \\ f_* \end{bmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} K & K_* \\ K_*^T & K_{**} \end{bmatrix}\right)$$



Back to introductory example (1)

Using the following Kernel : $k(a,b) : \exp\left(-\frac{1}{2}\frac{(a-b)^2}{l}\right)$ with $l = 100$

	x =20.0	x =40.0	x =30.0
x =20.0	1.00	0.14	0.61
x =40.0	0.14	1.00	0.61
x =30.0	0.61	0.61	1.00

K

	x =20.0	x =40.0
x =20.0	1.00	0.14
x =40.0	0.14	1.00

K_*

	x =30.0
x =20.0	0.61
x =40.0	0.61

K_{**}

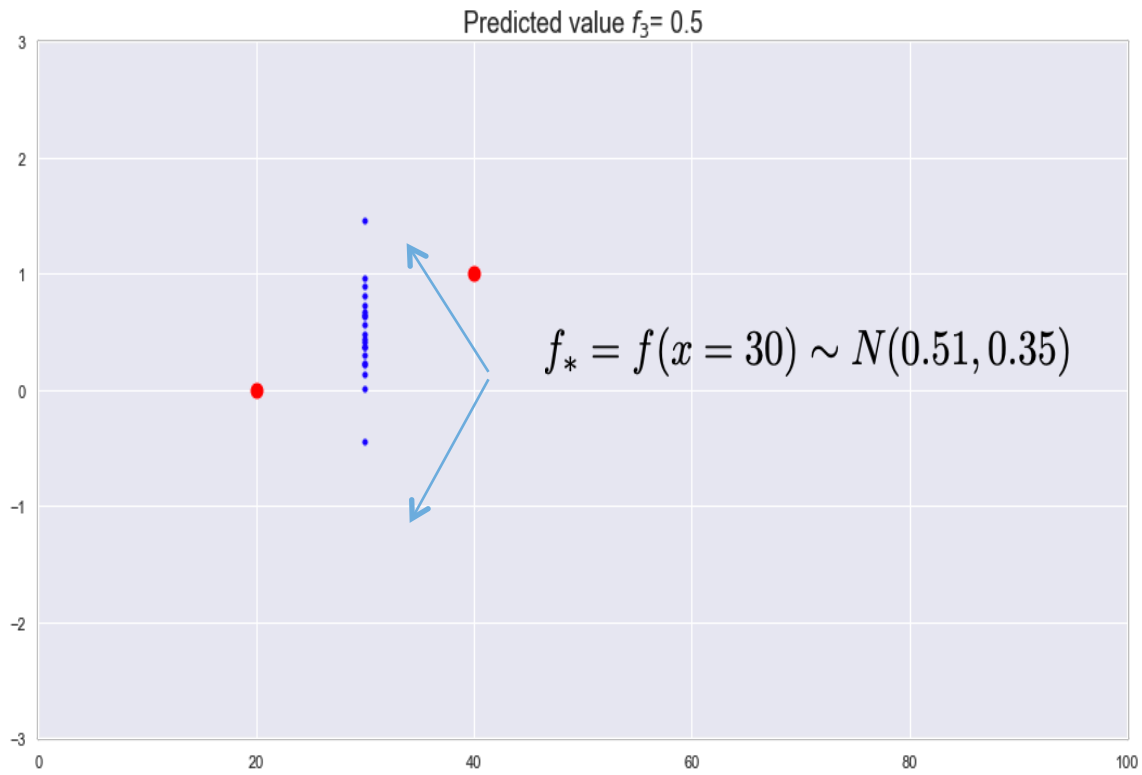
	x =30.0
x =30.0	1.0

$$\begin{aligned} \mu &= K_*^T K^{-1} f \\ \Sigma &= K_{**} - K_*^T K^{-1} K_* \end{aligned}$$

$\mu = 0.51$ and $\Sigma = 0.35$

$f_* = f(x = 30) \sim N(0.51, 0.35)$

Back to introductory example (2)



More points to predict

	x =20.0	x =30.0	x =40.0	x =50.0	x =60.0	x =70.0	x =80.0	x =90.0
x =20.0	1.00	0.61	0.14	0.01	0.00	0.00	0.00	0.00
x =30.0	0.61	1.00	0.61	0.14	0.01	0.00	0.00	0.00
x =40.0	0.14	0.61	1.00	0.61	0.14	0.01	0.00	0.00
x =50.0	0.01	0.14	0.61	1.00	0.61	0.14	0.01	0.00
x =60.0	0.00	0.01	0.14	0.61	1.00	0.61	0.14	0.01
x =70.0	0.00	0.00	0.01	0.14	0.61	1.00	0.61	0.14
x =80.0	0.00	0.00	0.00	0.01	0.14	0.61	1.00	0.61
x =90.0	0.00	0.00	0.00	0.00	0.01	0.14	0.61	1.00

K

	x =20.0	x =40.0
x =20.0	1.00	0.61
x =40.0	0.61	1.00

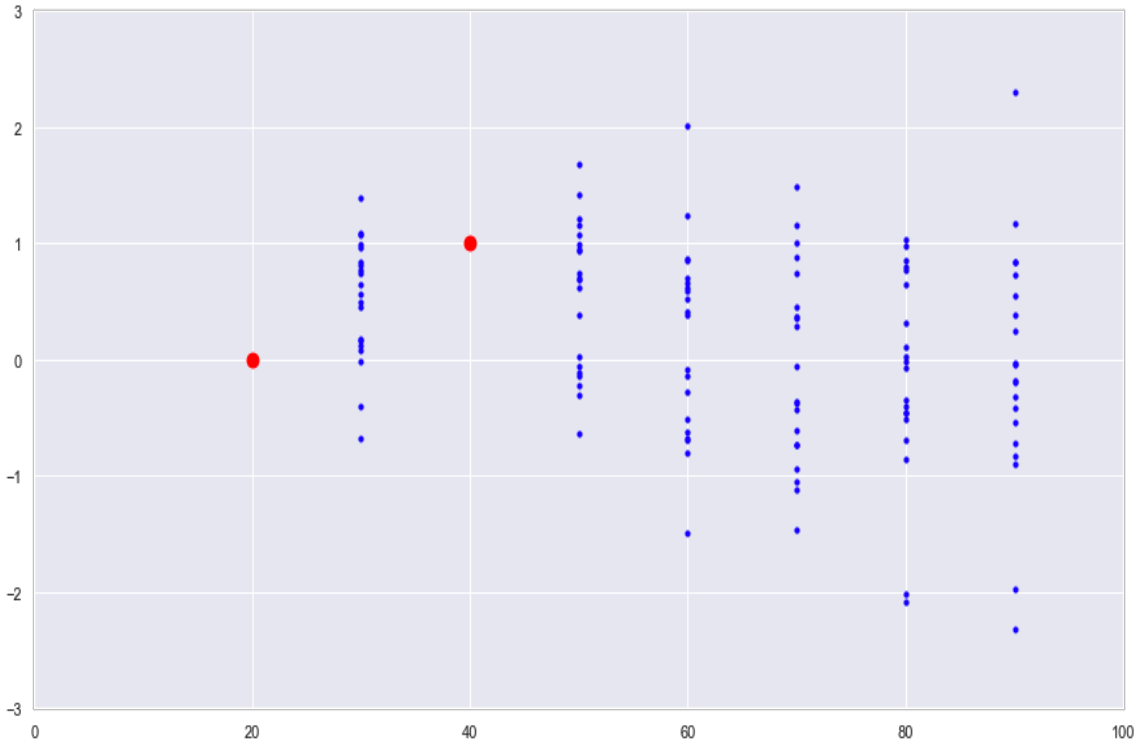
K_*

	x =30.0	x =50.0	x =60.0	x =70.0	x =80.0	x =90.0
x =20.0	0.14	0.01	0.00	0.0	0.0	0.0
x =40.0	0.61	0.14	0.01	0.0	0.0	0.0

K_{**}

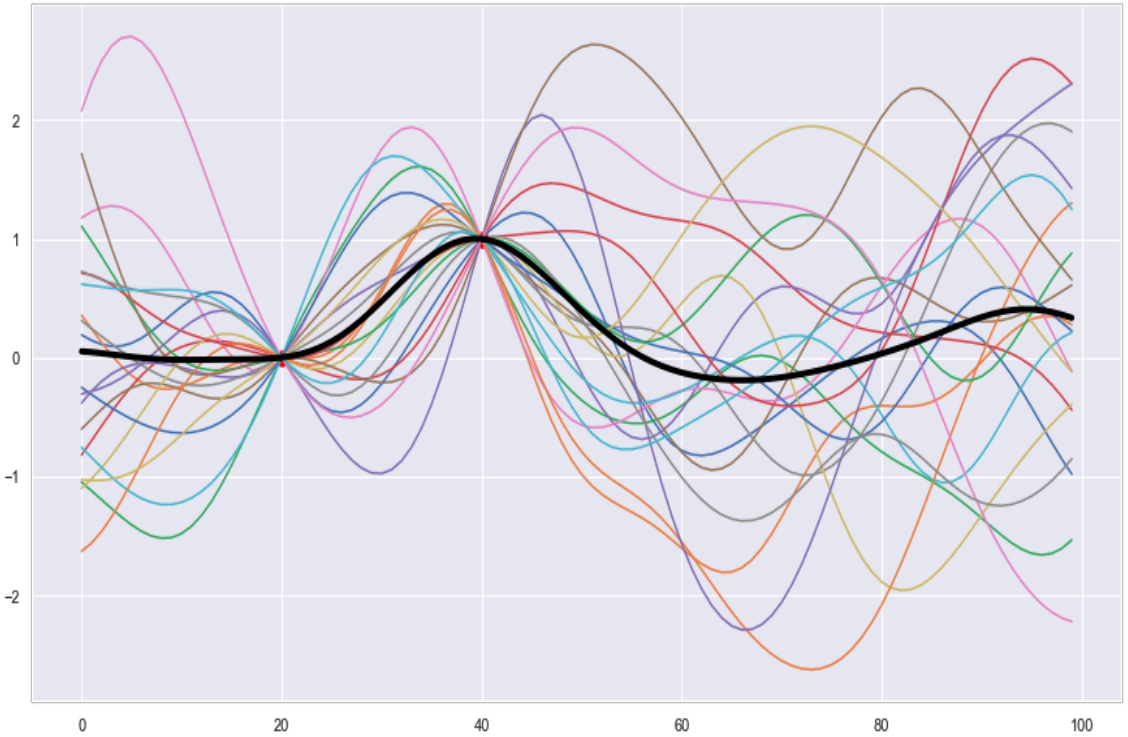
	x =30.0	x =50.0	x =60.0	x =70.0	x =80.0	x =90.0
x =30.0	1.00	0.61	0.14	0.01	0.00	0.00
x =50.0	0.61	1.00	0.61	0.14	0.01	0.00
x =60.0	0.14	0.61	1.00	0.61	0.14	0.01
x =70.0	0.01	0.14	0.61	1.00	0.61	0.14
x =80.0	0.00	0.01	0.14	0.61	1.00	0.61
x =90.0	0.00	0.00	0.01	0.14	0.61	1.00

More points to predict

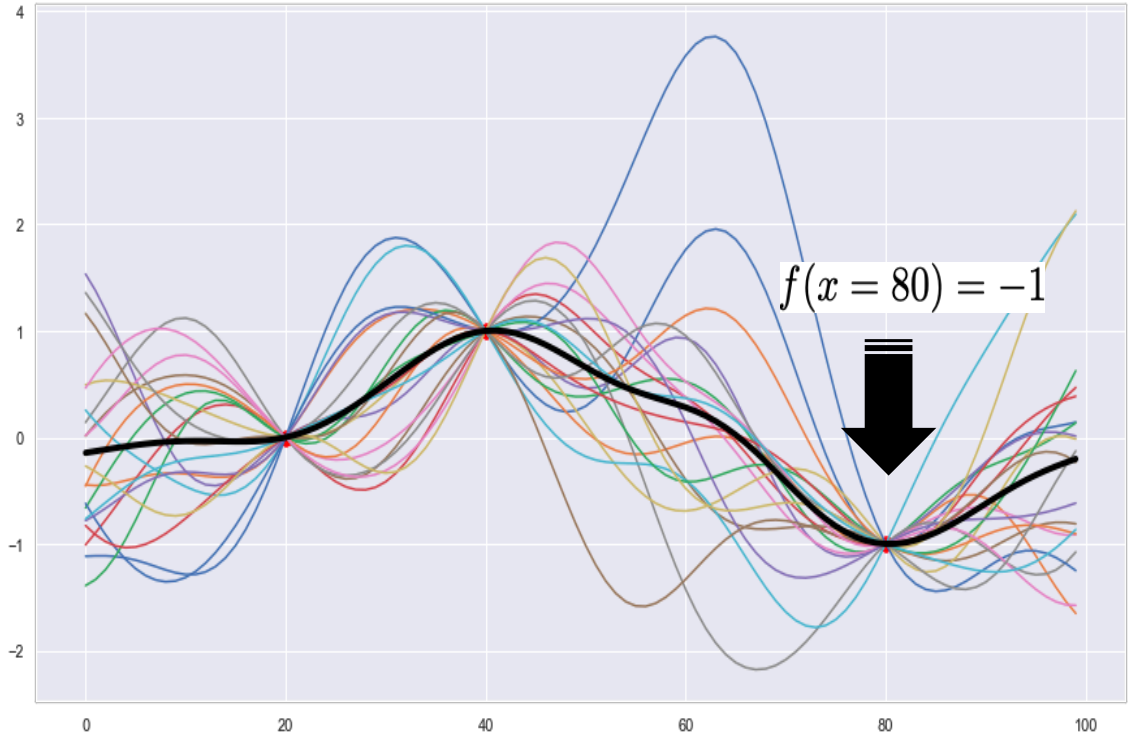


$$f_* \mid_{x_*, x, f} \sim N(\mu, \Sigma)$$

More points to predict



More points to predict + Extra point in training set



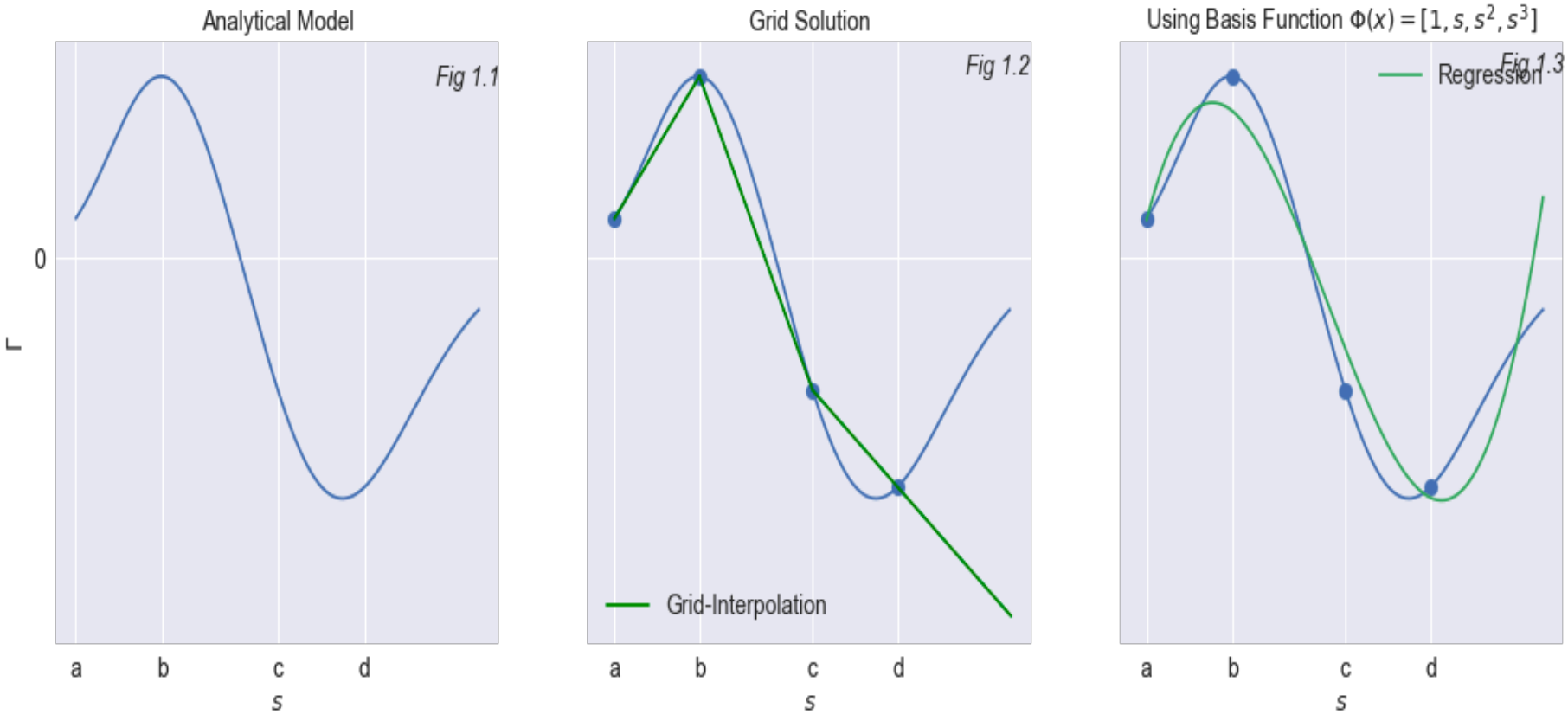
Where is the caveat?

$$\begin{aligned} \mu &= K_*^T K^{-1} f \\ \Sigma &= K_{**} - K_*^T K^{-1} K_* \end{aligned}$$

- The size of the matrix K grows with the training set
- Taking the inverse can be a computational challenge
- GPR is a technological challenge.

GPR in Derivatives Pricing

GPR Idea

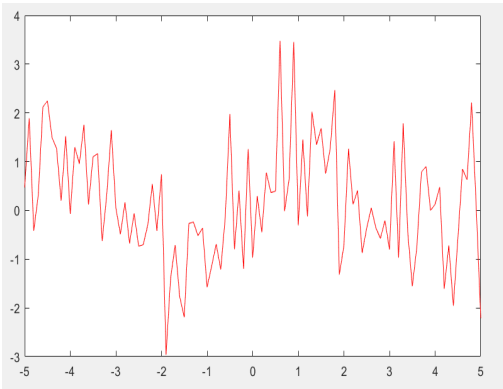

Analytical Model
Fig 1.1
0
r
a
b
c
d
s
Grid Solution
Fig 1.2
Grid-Interpolation
a
b
c
d
s
Using Basis Function $\Phi(x) = [1, s, s^2, s^3]$
Fig 1.3
Regression
a
b
c
d
s

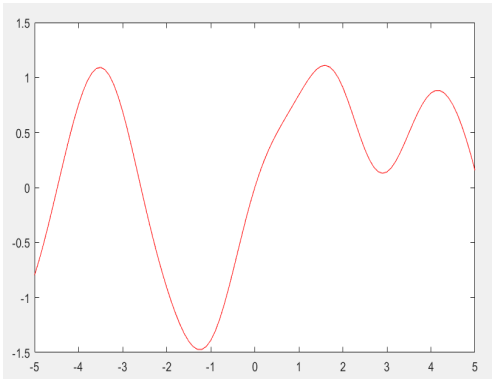
GPR Idea

- The GPR approach is a *non-parametric* approach, in that it finds a distribution over the possible **functions** $f(x)$ that are consistent with the observed data.
- As with all Bayesian methods it begins with a prior distribution and updates this as data points are observed, producing the posterior distribution over functions.

Not too wiggly

- *Not too wiggly*… what does that mean ?
- Here on the left is an example of a very wiggly function. And at the right here's a much smoother function:





- We use a **covariance matrix** to ensure that values that are close together in input space will produce output values that are close together. Covariance is a way to specify that smoothness:

low covariance = no relatio ⟷ high covariance = similar / close together

- This covariance matrix, along with a mean function to output the expected value of $f(x)$ defines a Gaussian Process.

GPR Definition

Consider a training set

$$(X, \boldsymbol{y}) = \{(\boldsymbol{x_i}, y_i) \mid i = 1, \ldots, n\}.$$

Find a relation between inputs and outputs:

$$y_i = f(\boldsymbol{x_i}) + \varepsilon_i$$

where $f(\boldsymbol{x})$ is a Gaussian process and $\varepsilon_i \sim \mathcal{N}(0, \sigma_n^2)$ are i.i.d. random variables representing the noise in the data.

GP: a Bayesian method

Posterior distribution

Only consider functions that agree with the data.

- Take new inputs X_*, with corresponding (unknown) function values $\boldsymbol{f}_*$
- Joint distribution of training outputs and function values:

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{f}_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}\,,\, \begin{bmatrix} K(X,X)+\sigma_n^2 I & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix}\right)$$

GP: a Bayesian method

Posterior distribution

- Condition on the observations:

$$\boldsymbol{f}_* | X_*, X, \boldsymbol{y} \sim \mathcal{N}\Big(\mu, \Sigma\Big)$$

with

$$\mu = K(X_*, X)\big[K(X, X) + \sigma_n^2 I\big]^{-1} \boldsymbol{y}$$

$$\Sigma = K(X_*, X_*) - K(X_*, X)\big[K(X, X) + \sigma_n^2 I\big]^{-1} K(X, X_*)$$

GP Kernels

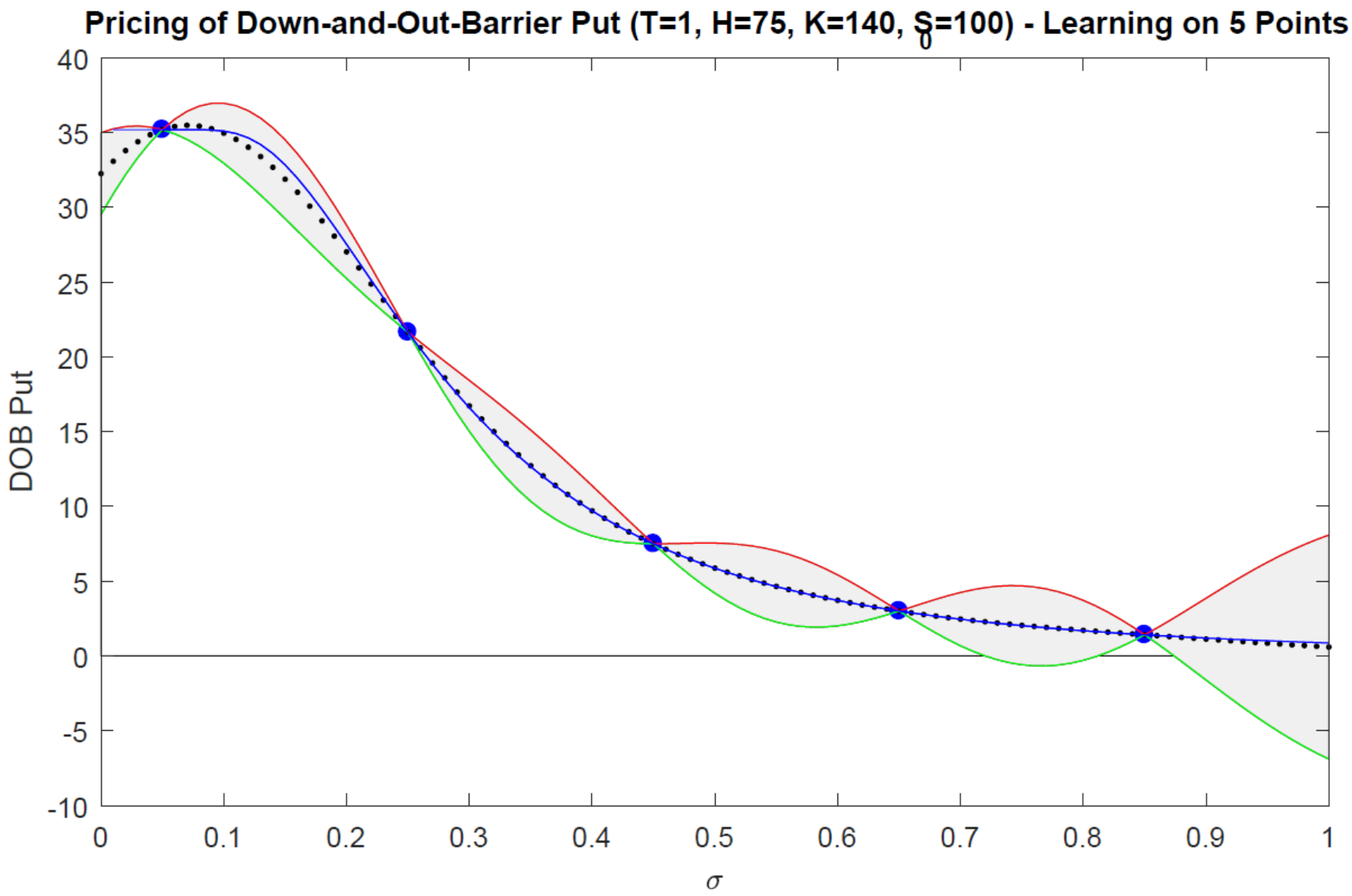
- Kernel functions : several choices are possible, each with different hyper-parameters

- How to determine the hyperparameters ?

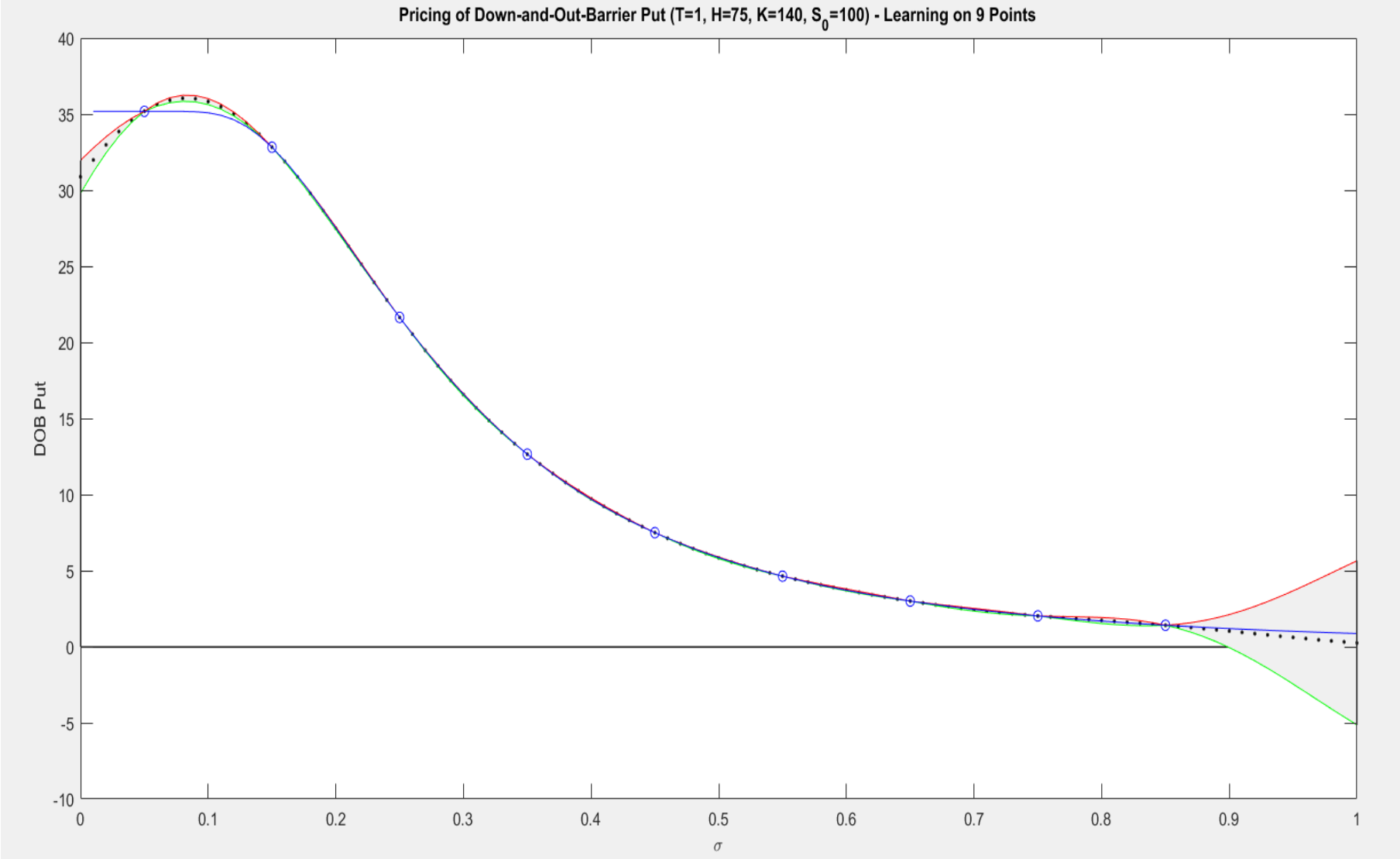
Maximize the marginal (log-)likelihood:

$$\boxed{-\frac{1}{2}\log(\det(K)) - \frac{1}{2}y^T K^{-1} y + const}$$

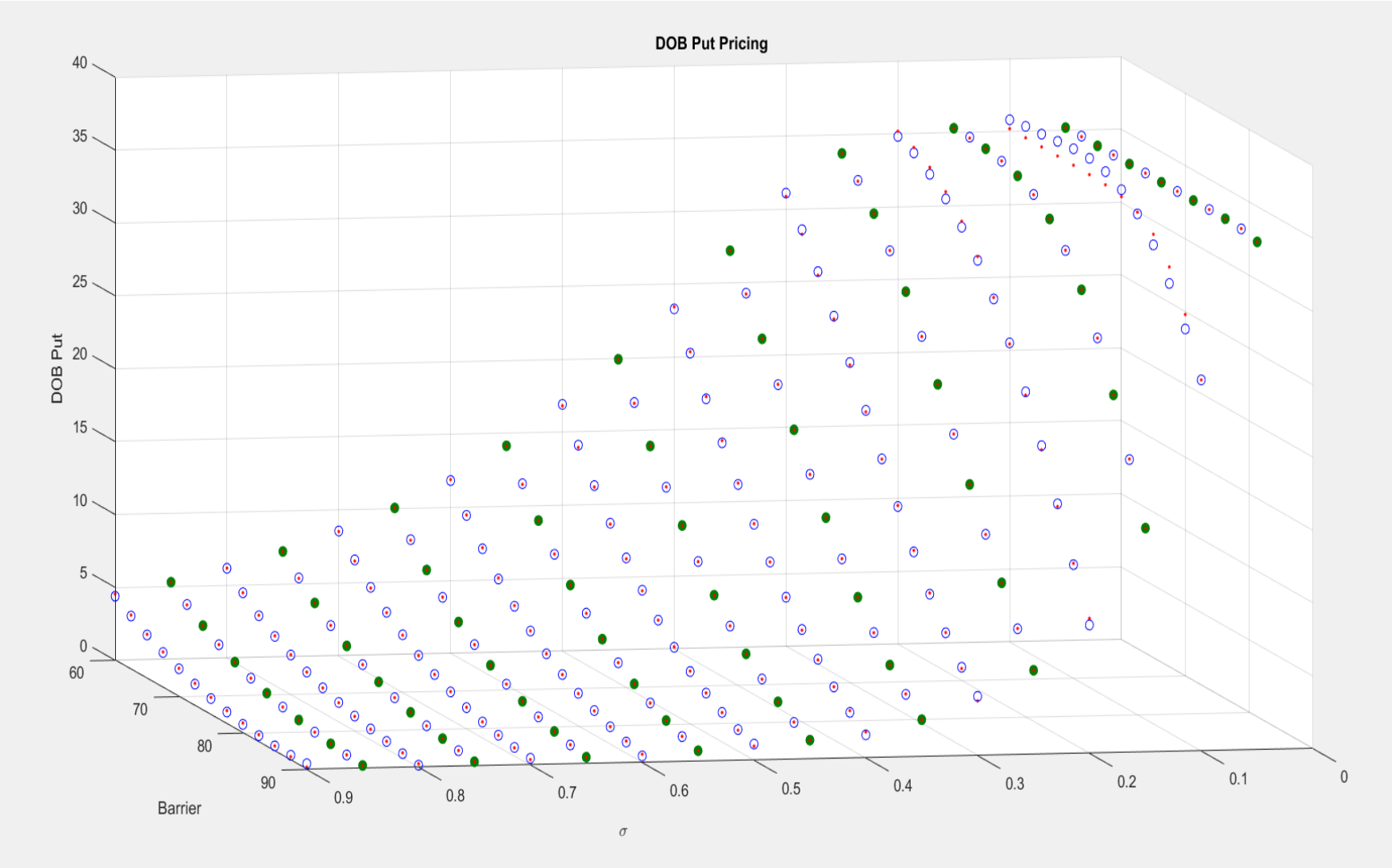
Barrier Option Pricing



Barrier Option Pricing



Barrier Option Pricing



Structured Products

Model: Heston

Training set:

- B : 1.05 → 1.55
- H : 0.55 → 0.95
- q : 0 → 0.05
- r : 0.02 → 0.03
- T : 11/12 → 1

- rho : -0.95 → -0.25
- v0 : 0.01 → 0.25
- kappa : 0.2 → 1.6
- theta : 0.15 → 0.65
- eta : 0.01 → 0.25

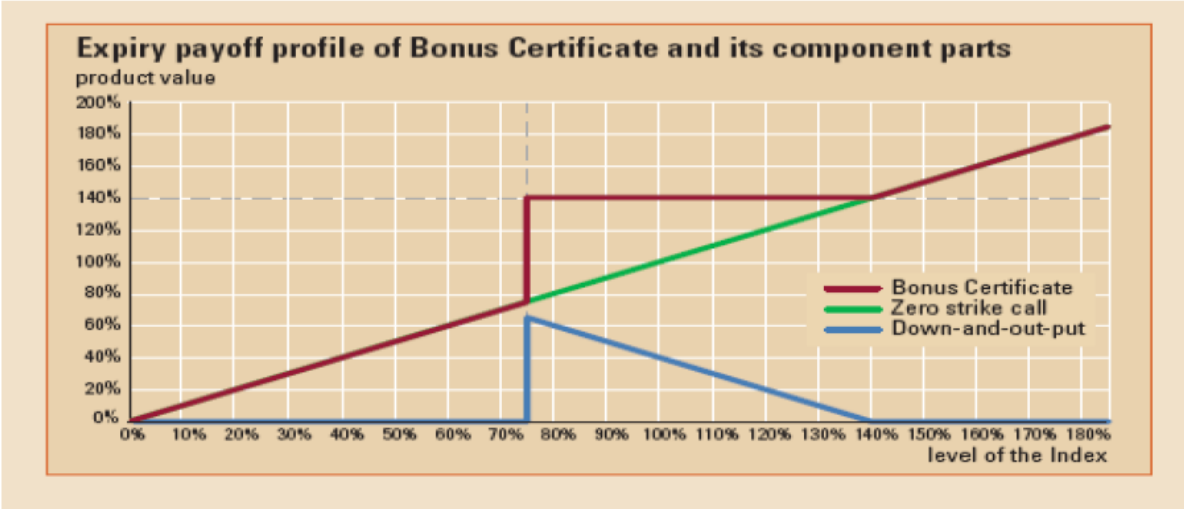
Assume $S_0 = 1$.

At expiry a holder recives

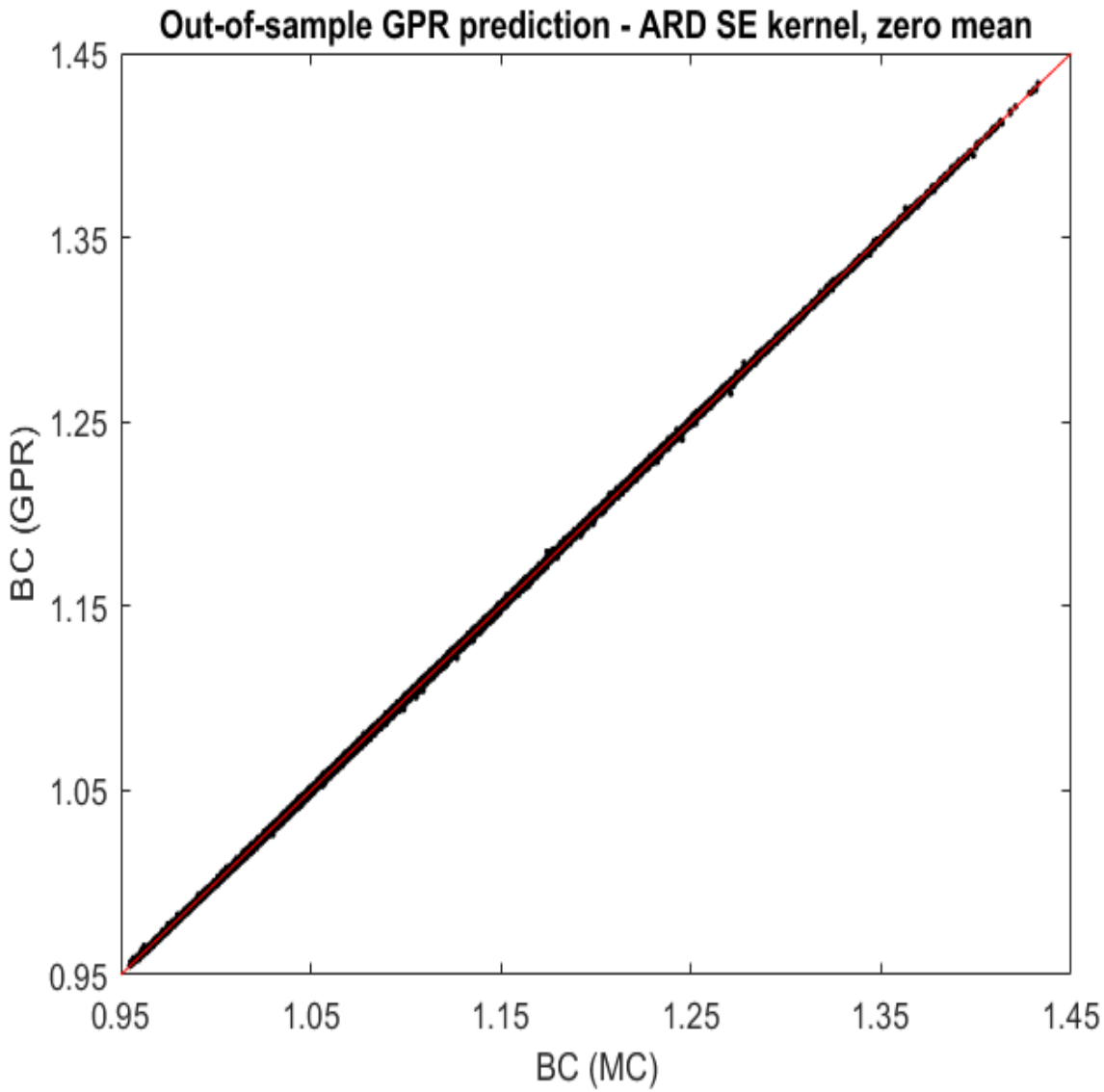
A. the final index level if it is above 140% of the initial level, or

B. 140% of the initial level, if the final level is between 75% and 140% of the initial level, unless the index level has fallen below 75% of the initial level during the lifetime of the certificate in which case…

C. one receives just the final level



Structured Products



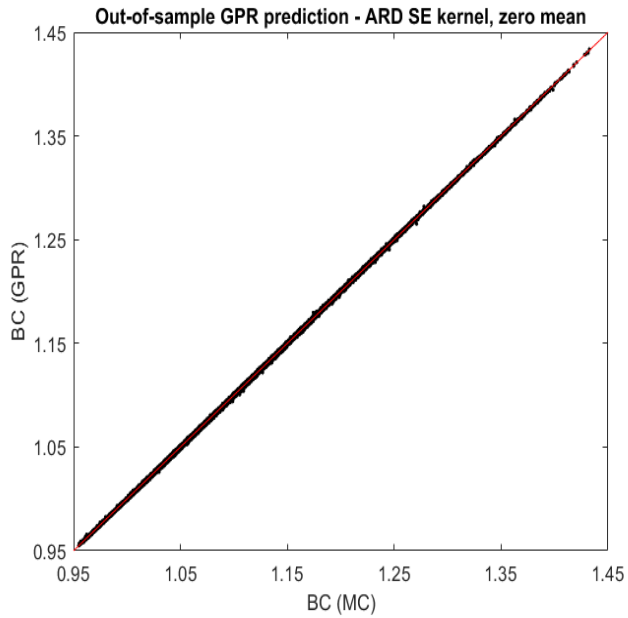
Performance

n_{train} = 10 000
n_{test} = 100 000

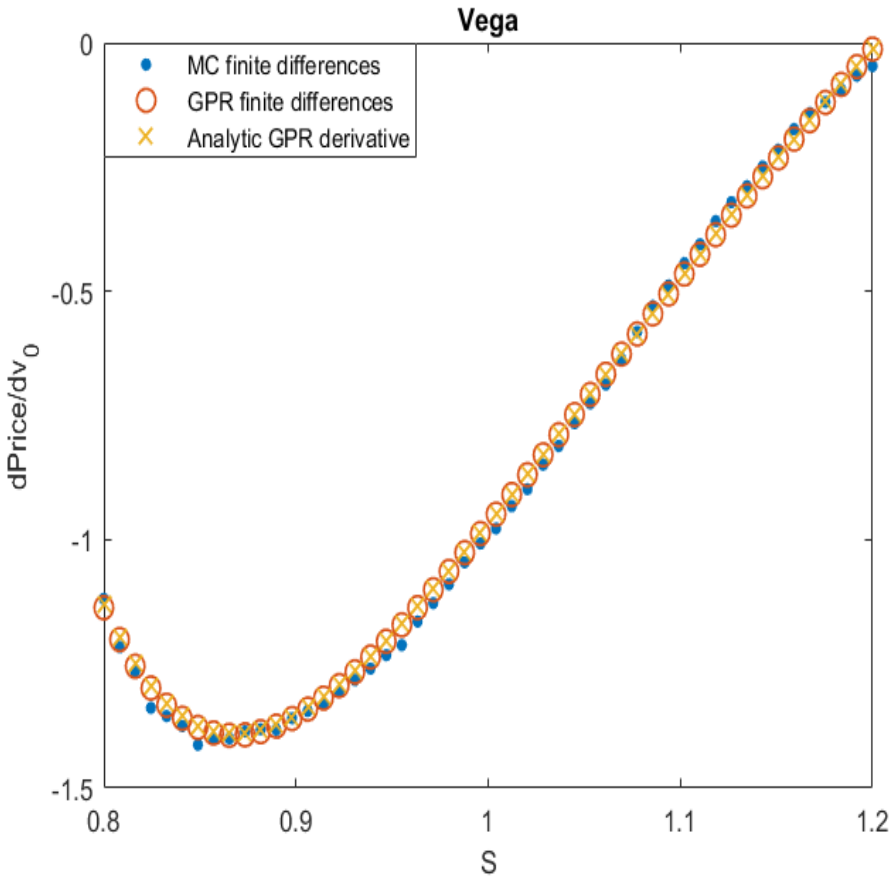
GPR Performance

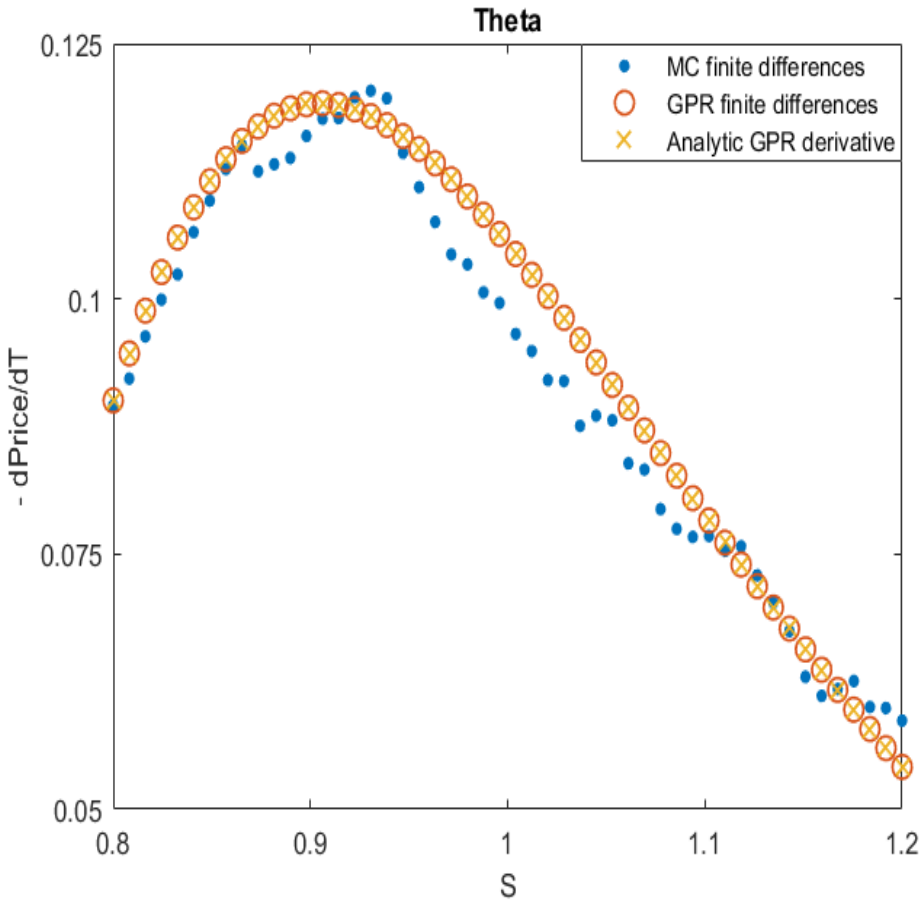
- Max. absolute error:
  0.0057
- Mean absolute error:
  0.0006
- Max. relative error:
  0.0045
- Mean relative error:
  0.0005

Speed-up: × 600



GPR: built-in Greeks





Reference : Crépey, Stéphane and Dixon, Matthew Francis, Gaussian Process Regression for Derivative Portfolio Modeling and Application to CVA Computations (January 30, 2019). Available at SSRN: https://ssrn.com/abstract=3325991 or http://dx.doi.org/10.2139/ssrn.3325991

Conclusion and more info

- We have presented new Machine Learning tools for solving traditional quant problems.
- Key is a fast matrix inversion of a huge kernel matrix.
- Speed-ups of several magnitudes can be achieved once the model is trained.

RiskConcile Solution: Plug-and-Price

Contact: wim@schoutens.be, jds@riskconcile.com

RISK CONCILE

Referenc De Spiegeleer, J., Madan, D. B., Reyners, S. and Schoutens, W. (2018), Machine Learning for Quantitative Finance: Fast Derivative Pricing, Hedging and Fitting, *Quantitative Finance*, forthcoming.