What is calibration?

- Derivative pricing models depend on parameters: Black-Scholes $\sigma$, interest rate $r$, Heston reversion speed $\kappa$, etc.
- For a given parameter $\theta$, one can compute model prices of derivatives, say, $P(\theta; S_0, t_0)$, conditional on the value of the underlying stock at time $t_0$, $S_0$.
- But: parameters are a priori unknown.
- Observe them indirectly via traded market prices $P^\star$.
- Calibration seeks to identify $\theta$ from the requirement

$$P(\theta; S_0, t_0) = P^\star,$$

where $P$ and $P^\star$ can be vectors.
Motivation

Time series analysis versus calibration to options

- Can use historical data to estimate model parameters.
- Asset price time series are observed in the physical measure.
- Option prices are observed in the risk-neutral measure (roughly: different drift, same vol).
- Option prices contain market expectation of future asset price moves.
- We focus solely on static calibration to quoted option prices.
- But see lectures on Time Series Analysis and Statistics.
Why calibrate?

- Calibration can be used to match the prices of liquidly traded contracts.
- It interpolates between prices, say for different strikes, maturities;
- to discover the true (?) model,
- to price more ‘exotic’ derivatives.
- Perhaps most importantly, it allows us to use the model to translate prices into hedge parameters.
- ...
**Motivation**

**Demands:**

- Fast direct solver
  - will need to generate prices for many different strikes, maturities, as part of some sort of iteration;
  - ideally analytic or semi-analytic formulae should be available for calibration products,
  - e.g., Fourier-based methods for *affine* models

- Stable parameter sensitivities
  - search directions of solvers usually based on gradients
  - Monte Carlo parameter ‘bumping’ can be noisy → see Monte Carlo lectures on pathwise sensitivities etc.
  - use algorithmic differentiation, adjoint methods etc

- Good initial guess
  - solver may only converge locally
  - parameter ballpark may be unknown
  - use guess based on analytic (asymptotic) approximation
Motivation

Potential, more fundamental problems:

- May not be able to fit all observed prices:
  \[ \{ \theta : P(\theta; S_0, t_0) = P^* \} = \emptyset \]

- May not have enough observed prices to determine all parameters:
  \[ |\{ \theta : P(\theta; S_0, t_0) = P^* \}| > 1 \]

- Estimation may be unstable:
  \[ P^{-1}(P^*; S_0, t_0) \text{ not continuous} \]

Trade-off between goodness of fit and stability of estimators, Greeks etc.
Outline

▶ Black-Scholes model and implied volatility
▶ Parametric models, non-linear solvers and fitting
▶ Discrete time formulae: Derman-Kani trees
▶ Continuous time formulae
  ▶ Shimko and the probability density function
  ▶ Dupire local volatility and the adjoint problem
▶ Regularisation
  ▶ Inverse and ill-posed problems
  ▶ Regularisation techniques
  ▶ Calibration examples for local volatility
  ▶ Bayesian methods
▶ Extensions
The Black-Scholes model assumes geometric Brownian motion
\[
\frac{dS}{S} = \mu \, dt + \sigma \, dW
\]
with a constant volatility parameter \( \sigma \). The PDE for the no-arbitrage price of a European option in this model is
\[
\frac{\partial V}{\partial t} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS \frac{\partial V}{\partial S} - rV = 0.
\]
In the simplest case of a vanilla call, there is a terminal condition
\[
V(S, T) = \max(S - K, 0).
\]
This problem can be solved *analytically* and the solution is

\[ V(S, t) = S \, N(d_1) - e^{-r(T-t)} K \, N(d_2), \]

where

\[ d_1 = \frac{\log(S/K) + (r + \frac{1}{2}\sigma^2)(T-t)}{\sigma \sqrt{T-t}}, \quad d_2 = d_1 - \sigma \sqrt{T-t}. \]

This gives us a mapping

\[ V = V(S, t; K, T; \sigma, r) \]

from the asset price $S$ and the parameters to the option price.
Implied volatility

- For an externally given price \( V^* \), the (Black-Scholes) implied volatility is the (constant) volatility parameter \( \sigma^* \), which, if inserted in the Black-Scholes model, gives this price.

- This definition only makes sense if such a parameter value exists, and is unique. From

\[
\frac{\partial V}{\partial t} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS \frac{\partial V}{\partial S} - rV = -\sigma S^2 \Gamma
\]

with \( \mathcal{V}(S, T) = 0 \) it can be shown that for convex payoffs the implied volatility is unique.

- In the case of European calls this means inverting the Black-Scholes formula with respect to \( \sigma \).

- Moreover, the implied volatility is the same for calls and puts (put-call-parity).
Empirical evidence

- The volatility $\sigma$ is a parameter of the model for the stock (the Black-Scholes model), and not of the option contract.
- If we believe in the model, we should expect to get the same implied volatility independent of what strike and expiry the option has that we use to identify $\sigma$.

Implied volatility for S&P 500 index call options.

<table>
<thead>
<tr>
<th>$T/K$</th>
<th>85</th>
<th>90</th>
<th>95</th>
<th>100</th>
<th>105</th>
<th>110</th>
<th>115</th>
<th>120</th>
<th>130</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.175</td>
<td>0.190</td>
<td>0.168</td>
<td>0.133</td>
<td>0.113</td>
<td>0.102</td>
<td>0.097</td>
<td>0.120</td>
<td>0.142</td>
<td>0.169</td>
</tr>
<tr>
<td>0.425</td>
<td>0.177</td>
<td>0.155</td>
<td>0.138</td>
<td>0.125</td>
<td>0.109</td>
<td>0.103</td>
<td>0.100</td>
<td>0.114</td>
<td>0.130</td>
</tr>
<tr>
<td>0.695</td>
<td>0.172</td>
<td>0.157</td>
<td>0.144</td>
<td>0.133</td>
<td>0.118</td>
<td>0.104</td>
<td>0.100</td>
<td>0.101</td>
<td>0.108</td>
</tr>
<tr>
<td>0.940</td>
<td>0.171</td>
<td>0.159</td>
<td>0.149</td>
<td>0.137</td>
<td>0.127</td>
<td>0.113</td>
<td>0.106</td>
<td>0.103</td>
<td>0.100</td>
</tr>
<tr>
<td>1.000</td>
<td>0.171</td>
<td>0.159</td>
<td>0.150</td>
<td>0.138</td>
<td>0.128</td>
<td>0.115</td>
<td>0.107</td>
<td>0.103</td>
<td>0.099</td>
</tr>
<tr>
<td>1.500</td>
<td>0.169</td>
<td>0.160</td>
<td>0.151</td>
<td>0.142</td>
<td>0.133</td>
<td>0.124</td>
<td>0.119</td>
<td>0.113</td>
<td>0.107</td>
</tr>
<tr>
<td>2.000</td>
<td>0.169</td>
<td>0.161</td>
<td>0.153</td>
<td>0.145</td>
<td>0.137</td>
<td>0.130</td>
<td>0.126</td>
<td>0.119</td>
<td>0.115</td>
</tr>
</tbody>
</table>
It is evident that the volatility surface is not flat as predicted by the Black-Scholes model. We often see both a
- dependence on $K$, termed *smile* (e.g., FX markets) or *skew* (e.g., stock markets), and a
- dependence on $T$, referred to as *term structure*. 
Another *parametric* example, the Heston stochastic volatility model:

\[
\begin{align*}
    dS_t &= rS_t \, dt + \sqrt{Y_t} S_t \, dW_t^1, \\
    dY_t &= \kappa (\theta - Y_t) \, dt + \xi \sqrt{Y_t} \, dW_t^2,
\end{align*}
\]

where \( Y_t \) is the *variance process* of the asset. \( \xi > 0 \) governs the volatility of variance, and \( \kappa, \theta > 0 \) the mean reversion.

- The Wiener processes \( W_t^1, W_t^2 \) have correlation \( \rho \).
- A hedging argument gives the pricing PDE

\[
\frac{\partial V}{\partial t} + \frac{1}{2} \left[ S^2 Y \frac{\partial^2 V}{\partial S^2} + 2 \rho \gamma Y S \frac{\partial^2 V}{\partial S \partial Y} + \xi^2 Y \frac{\partial^2 V}{\partial Y^2} \right] + rS \frac{\partial V}{\partial S} + \kappa (\theta - Y) \frac{\partial V}{\partial Y} - rV = 0
\]
Heston model properties

- Under the Feller condition $2\kappa \theta > \xi^2$, the variance process stays strictly positive.
- The characteristic function

$$\phi(u, t) = \mathbb{E}[\exp(iu \log(S_t))]$$

is analytically known as

$$\phi(u, t) = \exp(iu(\log(S_0) + (r - q)t)) \times \exp\left(\theta \kappa \xi^{-2}((\kappa - \rho \xi iu + d)t) - 2\log((1 - ge^{dt})/(1 - g))\right) \times \exp\left(v_0 \xi^{-2}(\kappa - \rho \xi iu + d)(1 - ge^{dt})/(1 - ge^{dt})\right)$$

$$d = \sqrt{(\rho \xi u - \kappa)^2 + \xi^2(iu + u^2)}$$

$$g = (\kappa - \rho \xi iu + d)/(\kappa - \rho \xi iu - d)$$

- This allows for a semi-closed-from solution for calls as

$$C(K, T) = \exp(-\alpha \log K) \frac{\pi}{\alpha} \int_0^\infty \exp(-iv \log K)\varrho(v) \, dv$$

$$\varrho(v) = \frac{\exp(-rT)\phi(v - (\alpha + 1)i, T)}{\alpha^2 + \alpha - v^2 + i(2\alpha + 1)v}$$

($\alpha$ arbitrary but such that $\alpha + 1$st moment exists.)
Parametric models

- We now focus on such parametric models,
- with a finite number of parameters, i.e., a finite-dimensional vector $\theta$.
- Ideally, we want that

$$P(\theta; S_0, t_0) = P^*,$$

the observed price, for all quoted products.
- At the same time, we want $\theta$ low-dimensional ("parsimonious")
- May not be able to fit all observed prices,

$$\{ \theta : P(\theta; S_0, t_0) = P^* \} = \emptyset.$$
- Look for best-fit solution

$$|P(\theta; S_0, t_0) - P^*| \rightarrow \min.$$
Outline

▶ Newton in one dimension
▶ Newton in more than one dimensions
▶ Newton without derivatives
▶ Global Newton
▶ Constraints
▶ Discussion
Nonlinear problems arise in a variety of settings and shapes. Backing out an implied vol from a quoted option price is one example, the solution of a utility maximisation problem a more complex one. We hide the details by writing

\[ F(x) = 0 \]

with nonlinear \( F \).

\( F \) can be a multi-dimensional nonlinear differential operator or a real valued function of a single variable. We will see later which characteristics of \( F \) are crucial, but first look at an illustrative example.
Perhaps the most frequently solved nonlinear problem in the world is the following:

Given the market price $c_{K,T}^{\text{market}}$ of a European call option with strike $K$ and maturity $T$, find the (Black-Scholes) implied volatility, which, if inserted in the Black-Scholes formula, gives that price.

That is to say we solve

$$c(S, T, K, r, \sigma) - c_{K,T}^{\text{market}} = 0$$

for $\sigma$, where

$$c(S, T, r, \sigma) = SN(d_+) - Ke^{-rT} N(d_-),$$

$$d_\pm = \log(S/K) + (r \pm \frac{1}{2} \sigma^2) T \over \sigma \sqrt{T}.$$
Visual inspection

We choose $K = 0.25$, $T = 1$, $r = 0.05$, $t = 0$, $S = 0.25$.

Plot $F(x) = c(S, T, K, r, x) - c_{\text{market}}^{K,T}$ with $c_{\text{market}}^{K,T} = 0.03557813696496455$.

By zooming in we detect the solution $x^* = 0.3$

We observe that due to monotonicity there is always a unique implied vol for market prices within arbitrage bounds.
Bisection

Turn the graphical method into an algorithm:

From prior knowledge, e.g. inspection of the graph, we may know

\[ F(a_0) < 0 \land F(b_0) > 0 \quad \text{for} \quad a_0 < b_0. \]

If \( F \) is continuous, there is a solution \( x^* \) in \([a_0, b_0]\) with \( F(x^*) = 0 \).

Now shrink the interval iteratively.

For \( k = 0, 1, 2, \ldots \) choose \( x_k = (a_k + b_k)/2 \):

\[ \begin{align*}
&\text{if} \ F(x_k) < 0, \text{ set } a_{k+1} = x_k, \ b_{k+1} = b_k \\
&\text{if} \ F(x_k) > 0, \text{ set } a_{k+1} = a_k, \ b_{k+1} = x_k
\end{align*} \]

How many iterations are required for a prescribed accuracy \( \epsilon \)?

Clearly \( b_k - a_k = 2^{-k} \cdot (b_0 - a_0) \), and therefore

\[ |x^* - x_k| \leq 2^{-k} \cdot (b_0 - a_0). \]

So for the error to be bounded by \( \epsilon \) we need

\[ k \geq - \log_2 \epsilon + \log_2(b_0 - a_0). \]
Iterative methods generate a sequence $x_1, x_2, \ldots$ of approximations to the solution.

In a one-step recursion we can write

$$x_{k+1} = \Phi(x_k)$$

with a suitable starting point $x_0$. The goal is

$$\lim_{k \to \infty} x_k = x^*$$

where $x^*$ is a solution, $F(x^*) = 0$.

For continuous $\Phi$, $x^*$ is a fixed-point,

$$\Phi(x^*) = x^*.$$
Banach’s fixed-point theorem

The following holds:

If

$$|\Phi(x) - \Phi(y)| \leq q\|x - y\| \quad \text{with} \quad q < 1$$

i.e. $\Phi$ is a contraction, then the iteration converges.

In fact

$$|x_k - x^*| \leq q|x_{k-1} - x_k| \leq q^k\|x_0 - x^*\|,$$

$$|x_k - x^*| \leq \frac{q^k}{1 - q}|x_1 - x_0|,$$

$$|x_k - x^*| \leq \frac{q}{1 - q}|x_k - x_{k-1}|.$$  

This motivates two rules for the construction:

1. fixed-points of $\Phi$ correspond to zeros of $F$ (consistency)
2. $\Phi$ is a contraction with small contraction number (convergence, speed)
A simple example satisfying the first is

$$\Phi(x) = x - \tau F(x).$$

Obviously a solution $F(x^*) = 0$ satisfies $\Phi(x^*) = x^*$, i.e. is a fixed point of the iteration.

$\Phi$ is a contraction, since $F$ is increasing with bounded slope, and $\tau$ is sufficiently small.
We look at the above example (implied vol) with $\tau = 5$.

The first 10 iterates are

$0.2, 0.247258, 0.272246, 0.285404, 0.292326, 0.295965, 0.297879, 0.298885, 0.299414, 0.299692$

and the error

$0.1, 0.0527416, 0.0277542, 0.014596, 0.00767444, 0.00403481, 0.00212121, 0.00111516, 0.000586254, 0.0003082$, 

i. e. goes down with a factor of roughly 0.5.
Convergence behaviour

From top left: $\tau = 1$ (slow monotone convergence), $\tau = -5$ (divergence), $\tau = 15$ (non-monotone convergence), $\tau = 25$ (limit cycle).
The first 30 iterates for $\tau = 1$ are

0.2, 0.209452, 0.218016, 0.225774, 0.232801, 0.239164, 0.244927, 0.250145, 0.254869, 0.259147, 0.263019, 0.266525, 0.269699, 0.272572, 0.275172, 0.277527, 0.279658, 0.281587, 0.283333, 0.284914, 0.286345, 0.28764, 0.288812, 0.289873, 0.290834, 0.291703, 0.29249, 0.293202, 0.293847, 0.294431, 0.294959.

Convergence is very slow, because the contraction number is close to 1.

More generally we could have

$$x_{k+1} = \Phi_k(x_k, x_{k-1}, \ldots, x_1, x_0),$$
or a recursion of length $m$

$$x_{k+1} = \Phi(x_k, \ldots, x_{k-m+1}).$$
Newton-Raphson method

Geometric motivation:

Choose $x_1$ as the intersection of the tangent to $F$ at $x_0$ with the $x$-axis,

$$F(x_0) + (x - x_0)F'(x_0) = 0.$$
Newton-Raphson method

Algebraic motivation:

Expand $F$

$$F(x) = F(x_0) + F'(x_0) \cdot (x - x_0) + \mathcal{O}(|x - x_0|^2)$$

and solve the linearised problem

$$F(x_0) + F'(x_0) \cdot (x_1 - x_0) = 0.$$ 

Therefore if $F'(x_0) \neq 0$

$$x_1 = x_0 - \frac{F(x_0)}{F'(x_0)}.$$

Generally

$$x_{k+1} = \Phi(x_k)$$

with

$$\Phi(x) = x - \frac{F(x)}{F'(x)}.$$
Newton-Raphson method

Note that

\[ \Phi'(x) = \frac{F(x)F''(x)}{F'(x)^2} \]

and therefore

\[ \Phi'(x^*) = 0 \]

which gives superlinear (quadratic) convergence.

The expansion

\[ F(x) = F(x_0) + F'(x_0) \cdot (x - x_0) + O(|x - x_0|^2) \]

also makes sense for \( F : \mathbb{R}^n \to \mathbb{R}^n \), if \( F'(x_0) \), the Jacobi matrix, is interpreted as an element of \( \mathbb{R}^n \). Then

\[ x_{k+1} = x_k - F'(x_k)^{-1}F(x_k) \]

basically requires the solution of a linear system in each iteration.
Example

For the above example we know the vega

\[
\frac{\partial}{\partial \sigma} c(S, T, K, r, \sigma) = S \sqrt{T} N'(d_+) \]

Newton’s method with \( x_0 = 0.2 \) gives

0.2, 0.300753, 0.3, 0.3, 0.3, ...

with errors

0.1, −0.000753, −4.82 \cdot 10^{-9}, 3.33 \cdot 10^{-16}, ...

The number of valid digits doubles with each iteration.
What if we choose a bad starting value?

\[3, -2.7591, 3.4317, -5.2348, 80.464, -6.6852 \cdot 10^{351}\]

Look at \( F \) for large arguments.
Newton’s method is *locally convergent*:

Assume $F$ differentiable and

\[ |F'(x) - F'(y)| \leq \alpha |x - y| \]
\[ |F'(x_0)^{-1}| \leq \beta \]
\[ |F'(x_0)^{-1} F(x_0)| \leq \gamma \]

If

\[ \alpha \beta \gamma < \frac{1}{2}, \]

then Newton’s method converges to a zero $x^*$ of $F$.

Convergence is quadratic, i.e.

\[ |x_{k+1} - x^*| \leq C |x_k - x^*|^2. \]
Newton’s method converges fast, if the initial guess is sufficiently close to the solution.

Is there a suitable globalisation strategy?

Newton method requires knowledge of the derivative, which is often unknown.

Can one approximate the derivative numerically (inexact Newton methods)? How does this impact convergence?

How often do we need to recompute? Ideally we would increase approximation if close to the solution.

Do we need good approximation of full matrix? Perhaps only the action in direction of the solution is relevant (quasi-Newton methods). Relatively cheap update from previous iterate may be possible...
Aside: Gradient computation

- Nonlinear solvers require the gradient/Jacobian, and potentially Hessian.
- For some ways of computing sensitivities (finite differences, pathwise sensitivities,...) see Monte Carlo and Finite Difference lectures.
- Alternatively, automatic code differentiation (AD) tools are available, applying a massive chain rule to the algorithm.
- Expensive if large number of parameters.
- *Adjoint* (or *reverse mode*, as opposed to *forward mode*) differentiation is much more efficient if there is a large number of inputs and small number of outputs.
Inexact Newton methods

The simplified Newton method

\[ x_{k+1} = x_k - \frac{F(x_k)}{F'(x_0)}, \]

is convergent of first order.

The expansion

\[ F(x) = F(\bar{x}) + (x - \bar{x})F'(\bar{x}) + o(|x - \bar{x}|^2) \]

motivates to require

\[ \bar{B}s = y = F(x) - F(\bar{x}) \]

for \( s = x - \bar{x} \) and an approximation \( \bar{B} \) to \( F'(\bar{x}) \).

This is called a quasi-Newton condition and the fundamental principle behind this class of methods.

Christoph Reisinger
Calibration Lecture 1: Background and Parametric Models
Secant method

Geometrically, replace the tangent \( t_k(x) = f(x_k) + f'(x_k)(x - x_k) \) by the secant
\[
s_k(x) = f(x_k) + \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}(x - x_k).
\]

We choose \( x_{k+1} \) as the intersection with the axis, \( s_k(x_{k+1}) = 0 \), then
\[
x_{k+1} = x_k - \frac{f(x_k)(x_k - x_{k-1})}{f(x_k) - f(x_{k-1})}.
\]

The method still converges superlinearly with order \((1 + \sqrt{5})/2 \approx 1.618\).

For the implied vol example the iterates are
\[
0.2, 0.25, 0.300261, 0.3, 0.3, 0.3, \ldots
\]

with error
\[
0.1, 0.05, -0.00026096, 2.6206 \cdot 10^{-7}, 5.9824 \cdot 10^{-13}, 4.4409 \cdot 10^{-16}.
\]
We define approximations $B_k$ to $F'(x_k)$, satisfying

$$B_k(x_k - x_{k-1}) = B_k s_k = y_k = F(x_k) - F(x_{k-1}).$$

Clearly for $n > 1$ does not determine $B_k$ uniquely. We therefore have some freedom to choose $B_k$ in a way that it can be constructed relatively easily from $B_{k-1}$, and it is common to require

$$B_k z = B_{k-1} z \quad \forall z \perp x_k - x_{k-1}.$$

This uniquely determines $B_{k+1}$ as

$$B_{k+1} = B_k + \frac{(y_k - B_k s_k) s_k^T}{s_k^T s_k}.$$
The update is then given by

\[ x_{k+1} = x_k + s_k \]

where \( s_k = B_k^{-1}F(x_k) \).

The prevailing mantra in scientific computing – never to invert any matrices – may be ignored here. The algorithmic advantage of a rank-one update is the relatively cheap availability of the inverse of \( B_{k+1} \), say \( H_{k+1} \), given \( H_k \).

It can be verified easily that \( B_{k+1} \) is indeed invertible and

\[ H_{k+1} = H_k + \frac{(s_k - H_k y_k)s_k^T H_k}{s_k^T H_k y_k} \]

provided that \( s_k^T H_k y_k \neq 0 \).

This is the Sherman-Morrison formula.
Introduce a level function

\[ f(x) = \frac{1}{2} |F(x)|^2 = \frac{1}{2} F(x)^T F(x) \]

with associated level sets

\[ L(x) = \{ y : T(y) \leq T(x) \} . \]

The level function has the property

\[ f(x) = 0 \iff x = x^* \]
\[ f(x) > 0 \iff x \neq x^* \]

This suggests to impose a monotonicity criterion

\[ f(x_{k+1}) \leq f(x_k) . \]
Steepest descent

Intuitively choose the direction of steepest descent:

\[ s_k = -f'(x_k) = -F'(x_k)F(x_k) \]

This choice has the downhill property

\[ f(x_k + \tau s_k) < f(x_k), \quad 0 < \tau < \tau_{max}. \]

For a real descent method perform a line search, e.g. find

\[ \tau_k = \inf\{\tau \geq 0 : \Phi'(|\tau|) = 0\} \]

with \( \Phi(\tau) = f(x_k + \tau s_k) \) and then set

\[ x_{k+1} = x_k + \tau_k s_k. \]

This can be solved approximately e.g. by a bisection strategy.

The method is known to be slow ('zig-zagging').
The above Newton iteration for a 'bad' initial guess failed, because the linearisation was a bad approximation at the next iterate. In this spirit, we restrict admissible new iterates to a region around the current approximation, the so-called trust region. It is common to take a ball

\[ |x - x_k| \leq r_k. \]

We solve the constrained optimisation problem

\[ y_{k+1} = \arg \min_{|x - x_k| \leq r_k} m_k(x) = \arg \min_{|x - x_k| \leq r_k} |F(x_k) + F'(x_k)(x - x_k)|^2 \]

for a candidate \( y_{k+1} \) for \( x_{k+1} \)

\( y_{k+1} \) is accepted if the actually achieved decrease in the objective function is comparable to the one predicted by \( m_k \):

\[ x_{k+1} = \begin{cases} y_{k+1} & \text{if } \rho_k = \frac{f(x_k) - f(y_{k+1})}{m_k(x_k) - m_k(y_{k+1})} \geq \frac{1}{4} \\ x_k & \text{otherwise} \end{cases} \]
The convergence properties depend crucially on the choice of $r_k$. To 'trust' the next iterate, we use a similar criterion to above.

A popular updating rule is the following:

$$ r_{k+1} = \begin{cases} 
\frac{r_k}{4} & \text{if } \frac{f(x_k) - f(y_{k+1})}{m_k(x_k) - m_k(y_{k+1})} < \frac{1}{4} \\
\min(2r_k, r_{max}) & \text{if } \frac{f(x_k) - f(y_{k+1})}{m_k(x_k) - m_k(y_{k+1})} > \frac{3}{4} \\
r_k & \text{otherwise}
\end{cases} $$

If the decrease is below the expectations, $m_k$ appears to be an unacceptable approximation at $y_{k+1}$ and we shrink the trust region.

For a reduction close to or above our expectations we may feel optimistic as to increase the admissible step size.

For non-active restrictions Newton’s method is retained.

Solve constrained problem using Lagrangian multipliers.
Implied vol, same data as before, $x_0 = 3 \ (x^* = 0.3)$. Conservative choice $r_0 = 0.1$. $r_{\text{max}} = 5$. The first 8 steps of the iteration:

1. $r_k = 0.1, \ y_{k+1} = 2.9, \ \rho_k = 1.03766, \ x_{k+1} = y_{k+1}$
2. $r_k = 0.2, \ y_{k+1} = 2.7, \ \rho_k = 1.07272, \ x_{k+1} = y_{k+1}$
3. $r_k = 0.4, \ y_{k+1} = 2.3, \ \rho_k = 1.13223, \ x_{k+1} = y_{k+1}$
4. $r_k = 0.8, \ y_{k+1} = 1.5, \ \rho_k = 1.18949, \ x_{k+1} = y_{k+1}$
5. $r_k = 1.6, \ y_{k+1} = 0.083731794, \ \rho_k = 0.963, \ x_{k+1} = y_{k+1}$
6. $r_k = 3.2, \ y_{k+1} = 0.328790844, \ \rho_k = 0.981, \ x_{k+1} = y_{k+1}$
7. $r_k = 5, \ y_{k+1} = 0.299998876, \ \rho_k = 1, \ x_{k+1} = y_{k+1}$
8. $r_k = 5, \ y_{k+1} = 0.300000000, \ \rho_k = 1, \ x_{k+1} = y_{k+1}$

The errors are

$-2.7, -2.6, -2.4, -2, -1.2, 0.21627, -0.028791, 1.1240 \cdot 10^{-6}, -1.07692 \cdot 10^{-14}$
Results

Bold choice $r_0 = 5$.

1. $r_k = 5, y_{k+1} = -2, \rho_k = -0.1649, x_{k+1} = 3$
2. $r_k = 1.25, y_{k+1} = 1.75, \rho_k = 1.432, x_{k+1} = y_{k+1}$
3. $r_k = 2.5, y_{k+1} = -0.080895778, \rho_k = 0.8978, x_{k+1} = y_{k+1}$
4. $r_k = 5, y_{k+1} = 0.402415974, \rho_k = 0.9374, x_{k+1} = y_{k+1}$
5. $r_k = 5, y_{k+1} = 0.29978676, \rho_k = 1, x_{k+1} = y_{k+1}$
6. $r_k = 5, y_{k+1} = 0.3, \rho_k = 1, x_{k+1} = y_{k+1}$

For points sufficiently close to the solution the method turns into the standard Newton method.
To generate good starting points, we turn the equation $F(x) = 0$ into a parametrised family

$$F(x, \tau) = 0, \quad \tau \in [0, 1]$$

such that $x_0$ is the solution for $\tau = 0$ (easy to solve problem) and $x^*$ is the solution for $\tau = 1$. A simple example is

$$F(x, \tau) = \tau F(x) + (1 - \tau)x_0.$$ 

If we choose sufficiently many steps in

$$F(x, \tau_k) = 0, \quad 0 = \tau_0 < \tau_1 < \ldots < \tau_N = 1,$$

then the solution to one problem will be a feasible initial point for Newton’s method applied to the next problem.

A related strategy embeds the equation in a dynamical system

$$\dot{x}(t) = F(x(t)) \quad x(0) = x(0)$$
Often, parameters are subject to constraints, e.g., Heston $\theta, \kappa$.

Then want to solve
\[
\min_{x \in \mathbb{R}^n} f(x), \quad l \leq x \leq u.
\]

One approach is penalisation, i.e., solve an unconstrained problem
\[
\min_{x \in \mathbb{R}^n} f(x) + g(x),
\]
where $g(x)$ “large” outside $l \leq x \leq u$.

The following method is due to Coleman and Li (”An interior trust region approach for nonlinear minimization subject to bounds”) and available in Matlab.
Define

\[ v_i(x) = \begin{cases} 
  x_i - u_i & \frac{\partial f}{\partial x_i} < 0 \text{ and } u_i < \infty \\
  -1 & \frac{\partial f}{\partial x_i} < 0 \text{ and } u_i = \infty \\
  x_i - l_i & \frac{\partial f}{\partial x_i} \geq 0 \text{ and } l_i > -\infty \\
  1 & \frac{\partial f}{\partial x_i} \geq 0 \text{ and } l_i = -\infty 
\end{cases} \]

Now consider

\[ \text{diag}(|v(x)|) \nabla f(x) = 0. \]

It is possible to construct a trust region Newton process, which

- generates a sequence of feasible points,
- is globally convergent, and
- of second order.