Optimisation Methods II

Matteo Fasiolo

Recap on SD and Newton's methods

Weaknesses of steepest descent:

- Provides direction but no information on step-length;
- Tends to zig-zag.

Improved by Newton's method via a better local model, but:

1. Needs to compute the Hessian matrix $\mathbf{H}$;
2. Need to ensure its positive definiteness.

Both need line search to guarantee convergence.

Can we avoid having to derive and compute $\mathbf{H}$?

Obvious solution is to approximate $\mathbf{H}$ via finite differences (FD).

This remove analytical effort, but still need to store $\mathbf{H}$ and solve:

$$\mathbf{H}\boldsymbol{\Delta} = -\nabla f(\boldsymbol{\theta}).$$

FD can be expensive: $p = \dim(\boldsymbol{\theta})$ gradient evaluations needed.

Alternative is to use **quasi-Newton** methods.

Idea is to use past gradients to update an approximate Hessian.

Update can be performed:

- Directly on inverse $\mathbf{B} = \mathbf{H}^{-1}$ so we can compute

$$\boldsymbol{\Delta} = -\mathbf{H}^{-1}\nabla f(\boldsymbol{\theta}) = -\mathbf{B}\nabla f(\boldsymbol{\theta}).$$

- So to guarantee that $\mathbf{B}$ is positive definite.

Quasi-Newton methods

At $\boldsymbol{\theta}^{[k]}$ we have pos. def. approx. to $\mathbf{H}^{[k]}$ and $\mathbf{B}^{[k]}$.

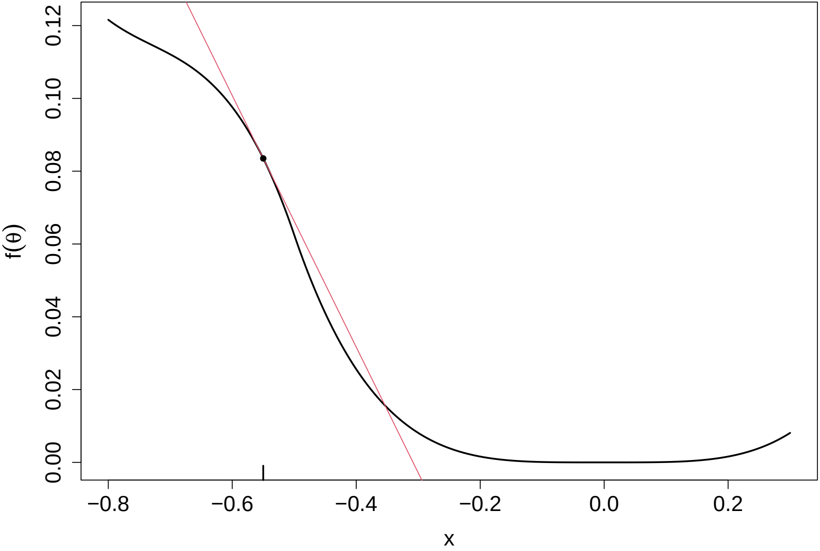
So search direction is

$$\boldsymbol{\Delta} = -\mathbf{B}^{[k]} \nabla f(\boldsymbol{\theta}^{[k]})$$

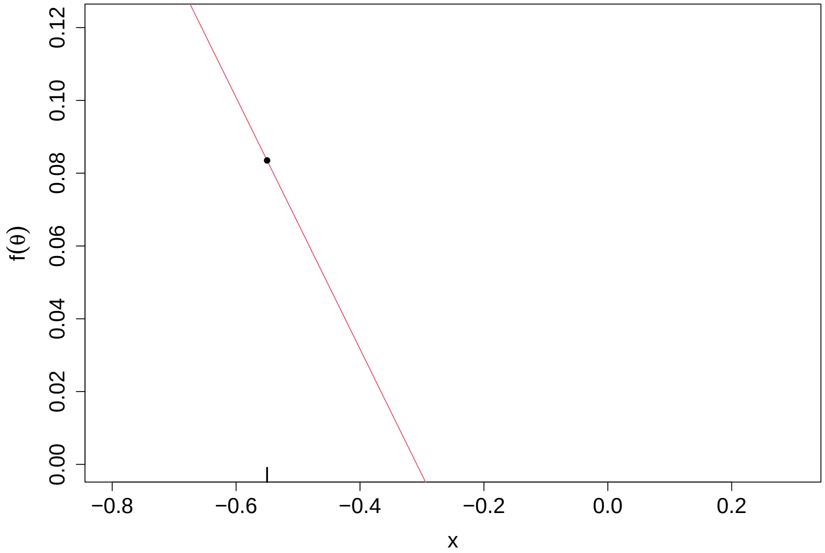
Along the $\boldsymbol{\theta}^{[k]} + \boldsymbol{\Delta}$ direction we can evaluate $f(\boldsymbol{\theta})$ and $\nabla f(\boldsymbol{\theta})$.

How to use them to update to (pos. def.) $\mathbf{H}^{[k+1]}$ and $\mathbf{B}^{[k+1]}$?

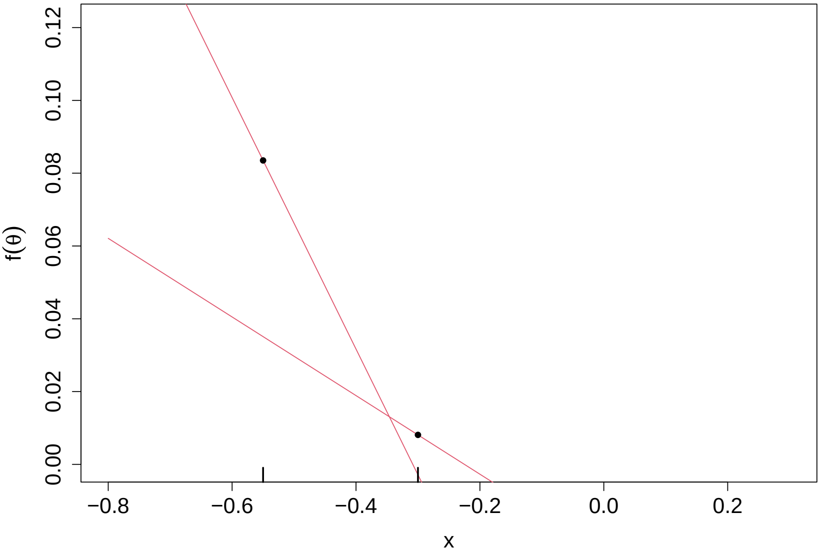
To get intuition, consider 1D case.



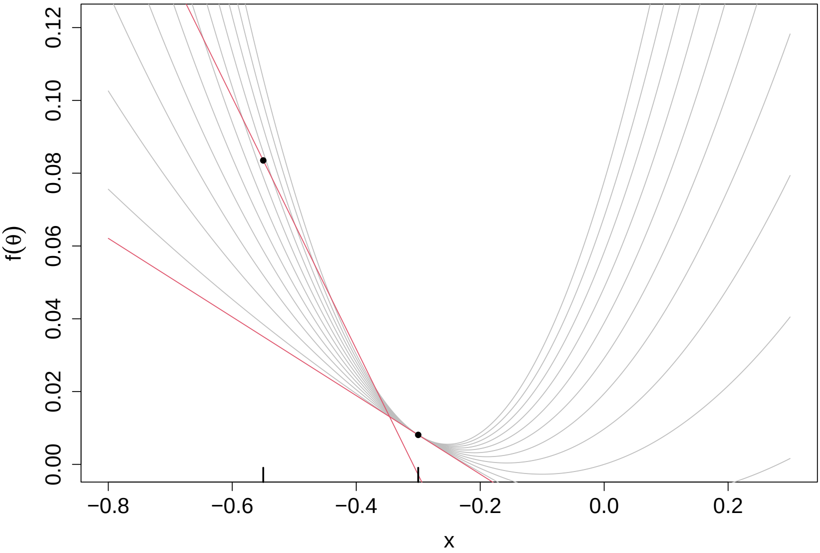
To get intuition, consider 1D case.



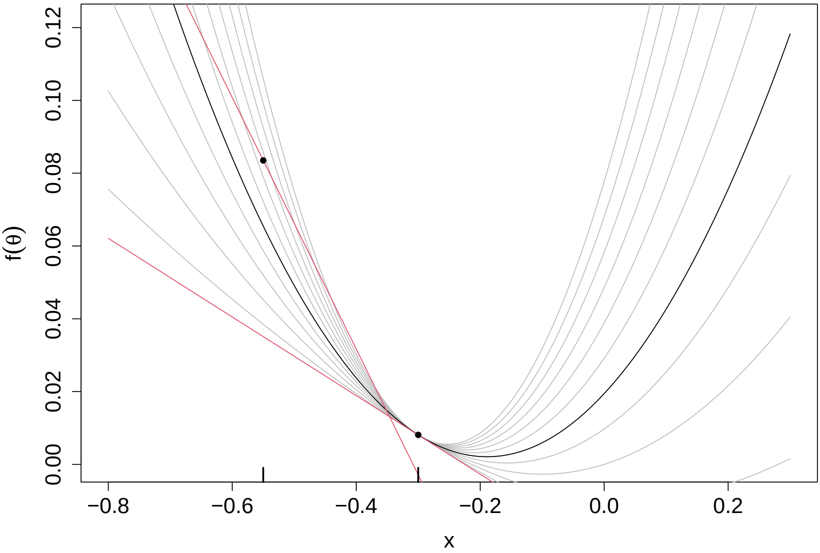
To get intuition, consider 1D case.



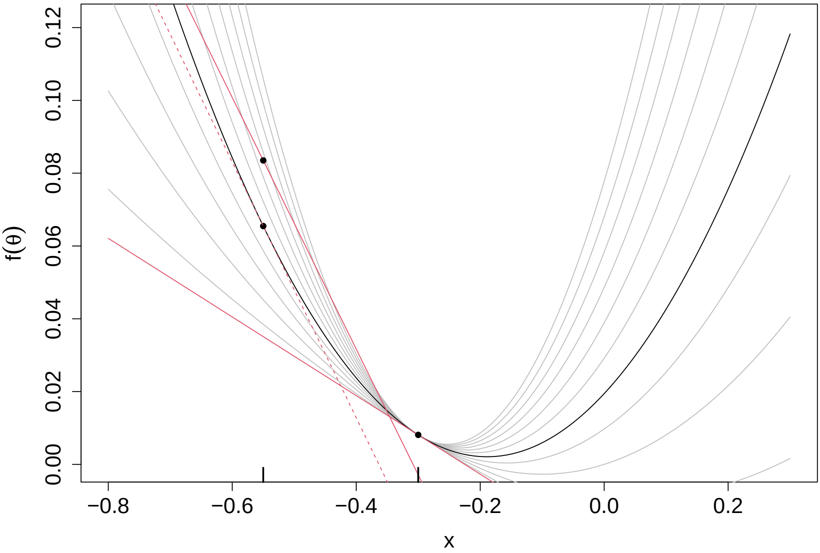
To get intuition, consider 1D case.



To get intuition, consider 1D case.



To get intuition, consider 1D case.



What are we doing? Let $f'(\theta)$ be $\frac{df}{d\theta}$.

We considered the local model at $\theta^{[k+1]}$

$$f(\theta) \simeq \tilde{f}(\theta) = f(\theta^{[k+1]}) + f'(\theta^{[k+1]})(\theta - \theta^{[k+1]}) + \frac{1}{2} H (\theta - \theta^{[k+1]})^2,$$

we differentiate w.r.t. θ

$$\tilde{f}'(\theta) = f'(\theta^{[k+1]}) + H(\theta - \theta^{[k+1]}),$$

and find H such that $f'(\theta^{[k]}) = \tilde{f}'(\theta^{[k]})$, that is

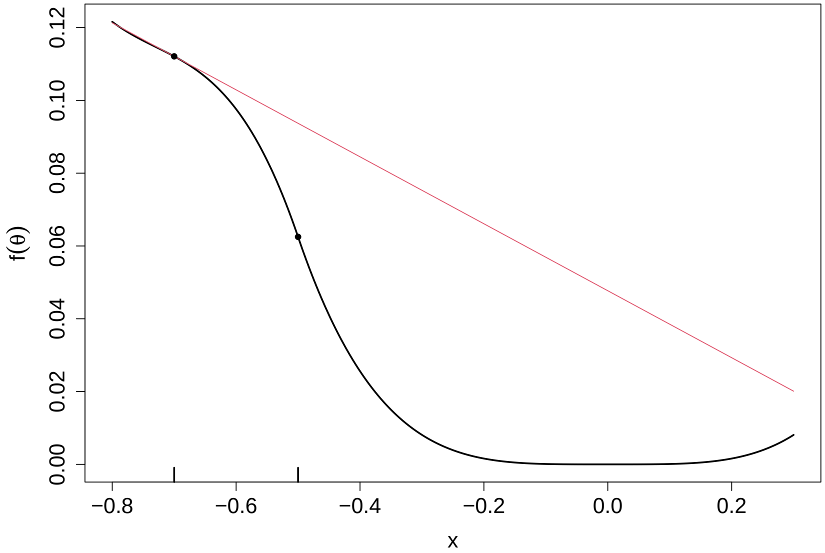
$$f'(\theta^{[k]}) = f'(\theta^{[k+1]}) + H(\theta^{[k]} - \theta^{[k+1]}),$$

so (anticlimax here)

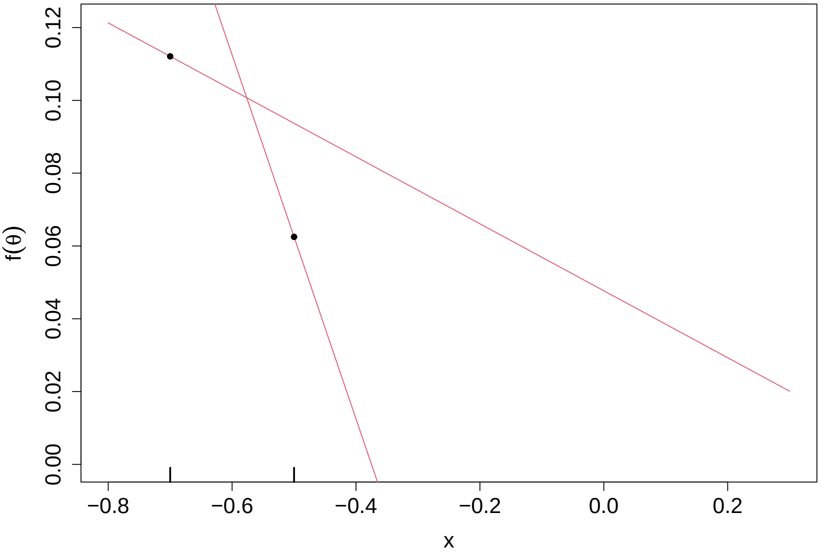
$$H^{[k+1]} = \frac{f'(\theta^{[k+1]}) - f'(\theta^{[k]})}{\theta^{[k+1]} - \theta^{[k]}} \quad \text{(finite differences, not updating } H^{[k]}\text{)}$$

But $H > 0$ implies that if $\theta^{[k+1]} > \theta^{[k]}$ then $f'(\theta^{[k+1]}) > f'(\theta^{[k]})$.

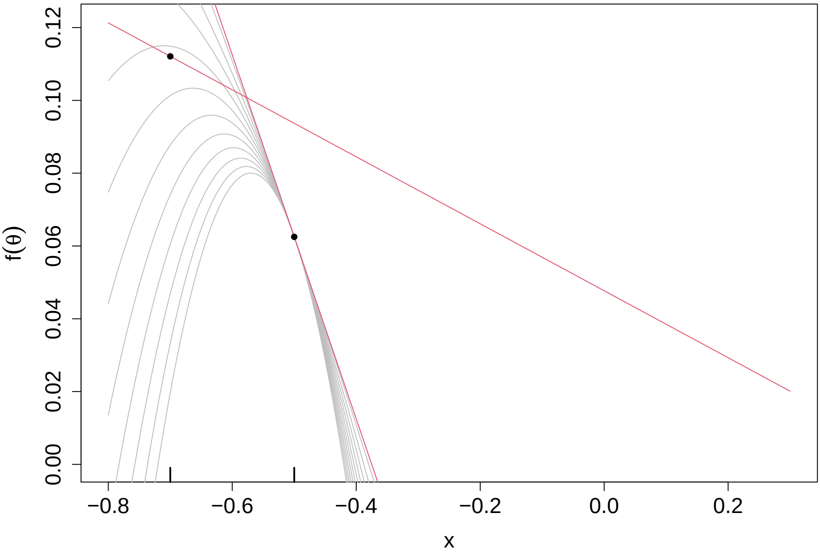
This is not always the case:



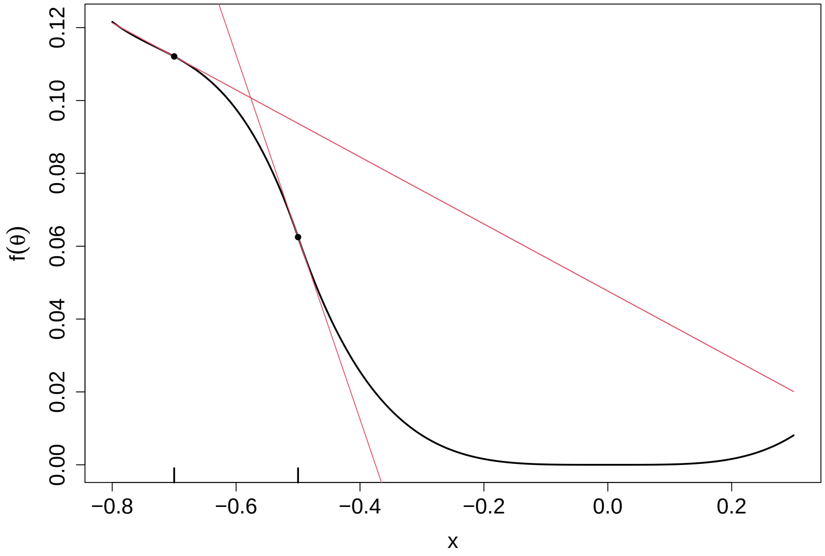
This is not always the case:



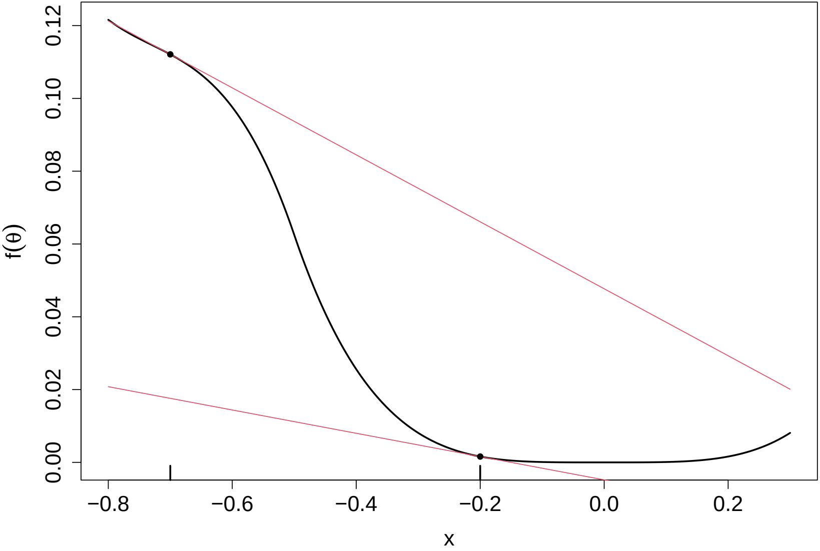
This is not always the case:



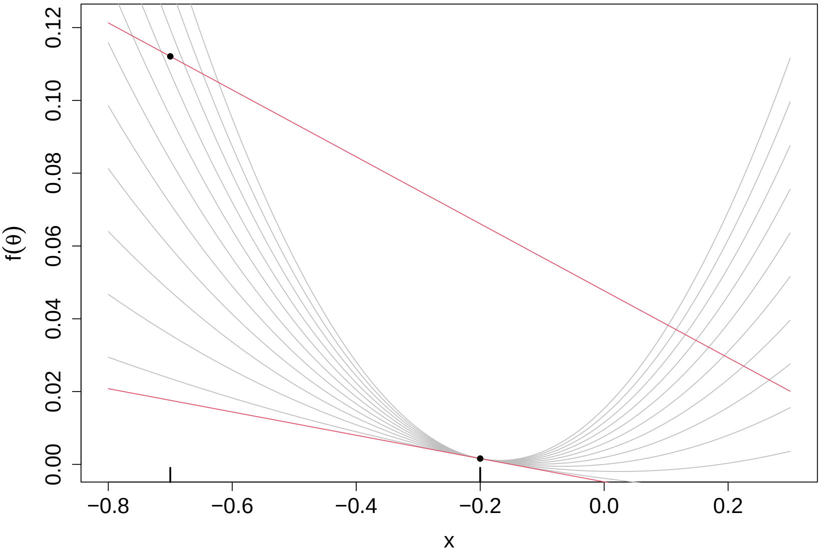
We need to increase the step-length:



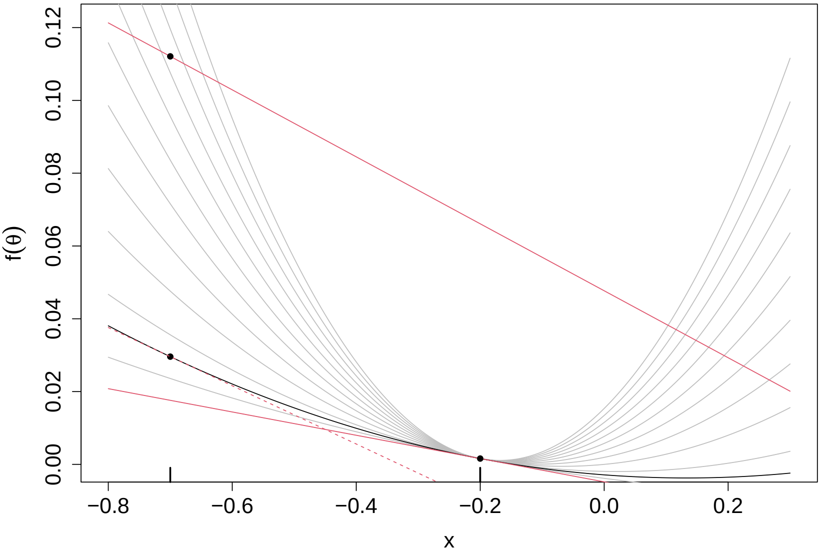
We need to increase the step-length:



We need to increase the step-length:



We need to increase the step-length:



In $\mathrm{D} > 1$ "gradient matching requirement" (**secant equation**) is

$$\nabla f(\boldsymbol{\theta}^{[k]}) = \nabla f(\boldsymbol{\theta}^{[k+1]}) + \mathbf{H}^{[k+1]}(\boldsymbol{\theta}^{[k]} - \boldsymbol{\theta}^{[k+1]}).$$

Rearranging

$$\nabla f(\boldsymbol{\theta}^{[k+1]}) - \nabla f(\boldsymbol{\theta}^{[k]}) = \mathbf{H}^{[k+1]}(\boldsymbol{\theta}^{[k+1]} - \boldsymbol{\theta}^{[k]}),$$

or $\mathbf{B}^{[k+1]}\mathbf{y}_k = \mathbf{s}_k$.

$\mathbf{H}^{[k+1]}$ has p^2 elements but equation imposes p constraints.

So it does not uniquely define $\mathbf{H}^{[k+1]}$ or its update from $\mathbf{H}^{[k]}$.

The BFGS method finds the update that solves

$$\mathbf{B}^{[k+1]} = \underset{\mathbf{B}}{\operatorname{argmin}} ||\mathbf{B} - \mathbf{B}^{[k]}||_{\mathrm{Frob}},$$

subject to

$$\mathbf{B} = \mathbf{B}^{\mathsf{T}} \quad \text{and} \quad \mathbf{B}\mathbf{y}_k = \mathbf{s}_k.$$

But there are other options, e.g. the DFP update for $\mathbf{H}^{[k+1]}$.

The solution to constrained optimisation problem is:

$$\mathbf{B}^{[k+1]} = (\mathbf{I} - \rho_k \mathbf{s}_k \mathbf{y}_k^\mathsf{T}) \mathbf{B}^{[k]} (\mathbf{I} - \rho_k \mathbf{y}_k \mathbf{s}_k^\mathsf{T}) + \rho_k \mathbf{s}_k \mathbf{s}_k^\mathsf{T}$$

where $\rho_k^{-1} = \mathbf{s}_k^\mathsf{T} \mathbf{y}_k$, or

$$\mathbf{B}^{[k+1]} = \mathbf{B}^{[k]} + \rho_k \mathbf{s}_k \mathbf{y}_k^\mathsf{T} \mathbf{B}^{[k]} \left(\rho_k \mathbf{y}_k \mathbf{s}_k^\mathsf{T} - 2\mathbf{I} \right) + \rho_k \mathbf{s}_k \mathbf{s}_k^\mathsf{T}$$

which is a rank-2 update.

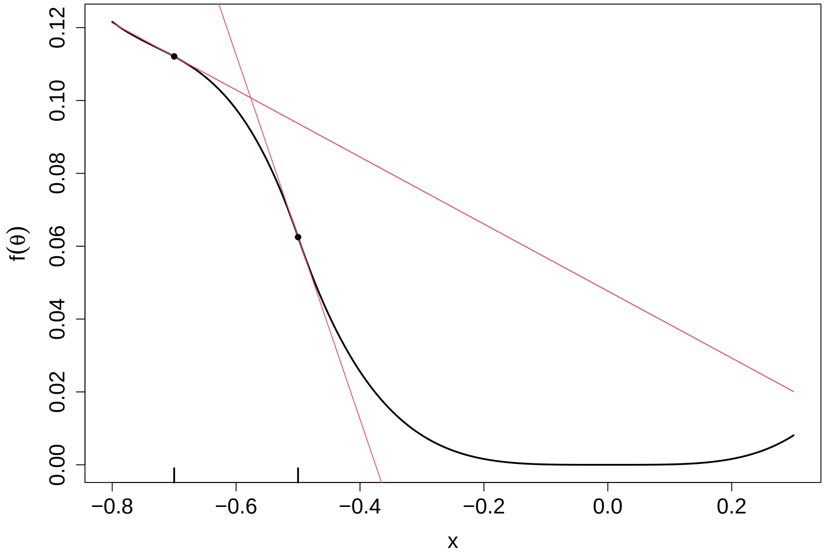
Note: we did not impose positive definiteness constraint on $\mathbf{B}^{[k+1]}$.

This is guaranteed if

$$\rho_k^{-1} = (\boldsymbol{\theta}^{[k+1]} - \boldsymbol{\theta}^{[k]})^\mathsf{T} (\nabla f(\boldsymbol{\theta}^{[k+1]}) - \nabla f(\boldsymbol{\theta}^{[k]})) > 0.$$

You want $f(\boldsymbol{\theta})$ to become flatter in the direction of the step.

Here ρ_k would be negative:



$\rho_k > 0$ ensured by step-length α statisfying **Wolf conditions**.

Recall step is $\boldsymbol{\Delta}_\alpha = \alpha\boldsymbol{\Delta} = -\alpha\mathbf{B}^{[k]}\nabla f(\boldsymbol{\theta}^{[k]})$.

Recall $\nabla f(\boldsymbol{\theta}^{[k]})^\mathsf{T}\boldsymbol{\Delta}_\alpha < 0$ if $\mathbf{B}^{[k]}$ is pos def.

First is **sufficient decrease** condition:

$$f(\boldsymbol{\theta}^{[k]} + \boldsymbol{\Delta}_\alpha) \leq f(\boldsymbol{\theta}^{[k]}) + c_1\nabla f(\boldsymbol{\theta}^{[k]})^\mathsf{T}\boldsymbol{\Delta}_\alpha$$

with $c_1 \in (0, 1)$.

So condition says that $f(\boldsymbol{\theta}^{[k]} + \boldsymbol{\Delta}_\alpha) < f(\boldsymbol{\theta}^{[k]})$ by a margin.

Then there is a **curvature condition**:

$$\nabla f(\boldsymbol{\theta}^{[k]} + \boldsymbol{\Delta}_\alpha)^\mathsf{T} \boldsymbol{\Delta}_\alpha \geq c_2 \nabla f(\boldsymbol{\theta}^{[k]})^\mathsf{T} \boldsymbol{\Delta}_\alpha,$$

with $c_2 \in (c_1, 1)$.

Substracting $\nabla f(\boldsymbol{\theta}^{[k]})^\mathsf{T} \boldsymbol{\Delta}_\alpha$ from both sides

$$(\nabla f(\boldsymbol{\theta}^{[k]} + \boldsymbol{\Delta}_\alpha) - \nabla f(\boldsymbol{\theta}^{[k]}))^\mathsf{T} \boldsymbol{\Delta}_\alpha \geq (c_2 - 1) \nabla f(\boldsymbol{\theta}^{[k]})^\mathsf{T} \boldsymbol{\Delta}_\alpha,$$

or

$$\rho_k^{-1} \geq (c_2 - 1) \nabla f(\boldsymbol{\theta}^{[k]})^\mathsf{T} \boldsymbol{\Delta}_\alpha > 0.$$

So it guarantees that $\mathbf{B}^{[k+1]}$ will be pos def.

Relative to Newton's method, with BFGS:

- You do not need to find the Hessian;
- You do not need to solve $\boldsymbol{\Delta}_\alpha = \mathbf{H}^{-1}\nabla f$ which is $O(p^3)$;
- Step will not be as good as Newton because you are approximating $\mathbf{H}$;
- You need to initialise $\mathbf{B}^{[0]}$.

Let's look at some examples:

```r
library(FLtools)
FLtools::optimisation()
```

What if I don't want to compute $\nabla f(\boldsymbol{\theta})$?

You can try the **Nelder-Mead** or **downhill symplex** optimiser.

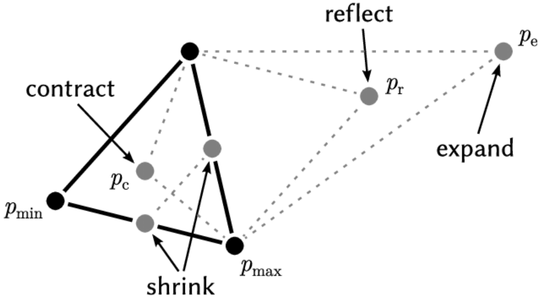


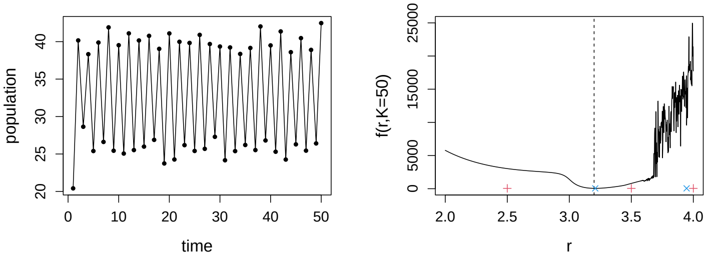
Figure 1: From Wikipedia

NOTE: it's the default optimiser in `stats::optim()`.

For examples, see `FLtools::optimisation()`.

Leading to the next part

We mentioned that it's important to look at your objective.

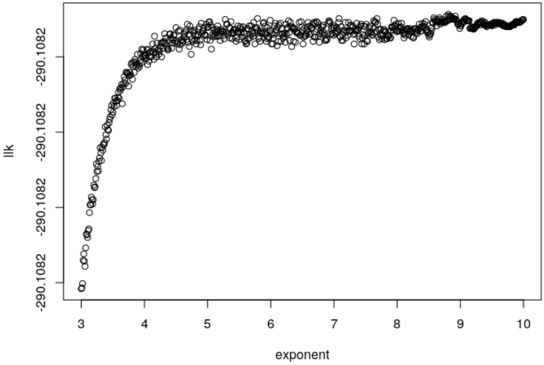


But in which direction should we look when $D \gg 1$?

SD, Newton and BGFS provide a search direction $\boldsymbol{\Delta}$.

That's the first direction along which to look.

Here is a real-world example I've encountered.



- y-axis is loglik($\boldsymbol{\theta} + \alpha \boldsymbol{\Delta}$)
- x-axis is γ used in

$$\alpha = \frac{1}{2^{\gamma}}$$

Objective is deterministic but evaluated up to **numerical noise**.