Nonlinear optimization

without constraints

Michel Bierlaire

Introduction to optimization and operations research

EPFL

Nonlinear optimization

Problem definition

$$\min_{x \in \mathbb{R}^n} f(x)$$

where f is twice differentiable and bounded from below:

$$\exists M \in \mathbb{R} \text{ such that } f(x) \geq M, \ \forall x \in \mathbb{R}^n.$$

Fermat's theorem

Theorem 5.1

- x^* is a local minimum of $f : \mathbb{R}^n \to \mathbb{R}$.
- If f is differentiable around x^*, then

$$\nabla f(x^*) = 0.$$

- If f is twice differentiable around x^*, then

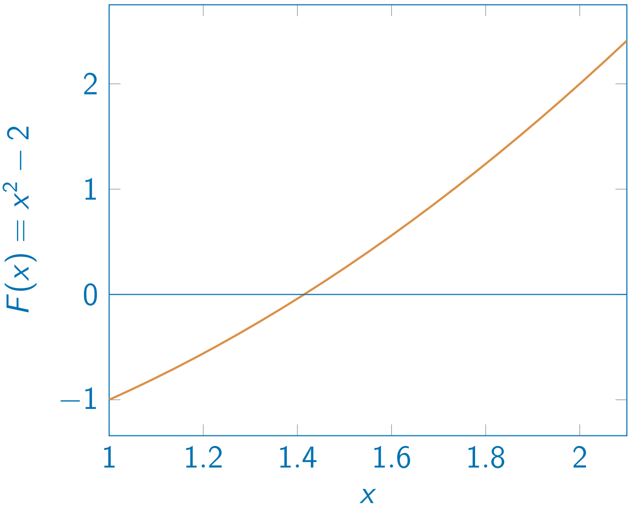
$$\nabla^2 f(x^*) \geq 0 \quad \text{[positive semidefinite]}.$$

Solving one equation with one variable

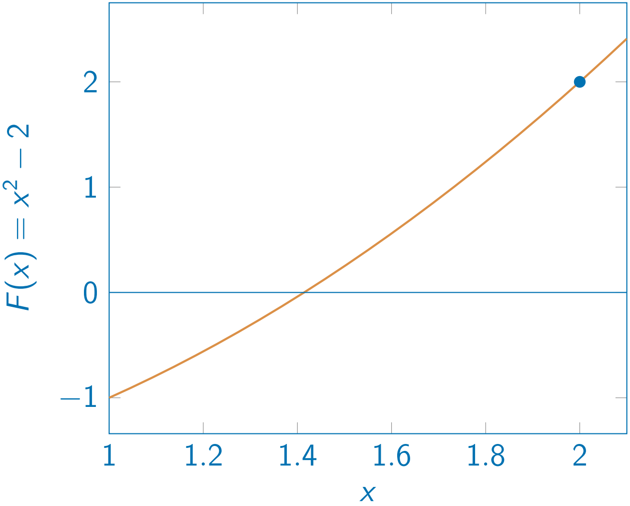
Motivation

- Optimization algorithms rely on solving systems of equations given by optimality conditions.
- Systems of linear equations can be solved by Gaussian elimination.
- Systems of nonlinear equations can be solved using Newton's method.
- We remind this method on the simple case of one equation and one variable.

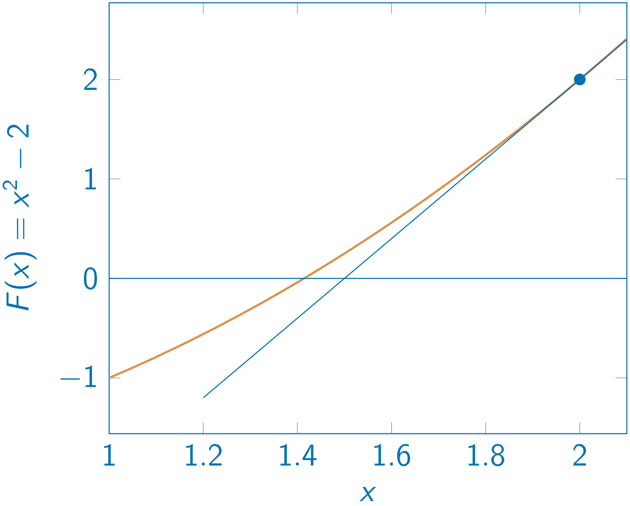
Example



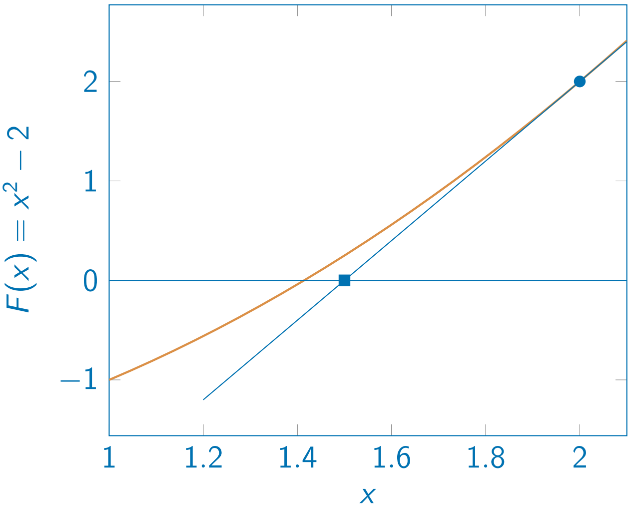
Example



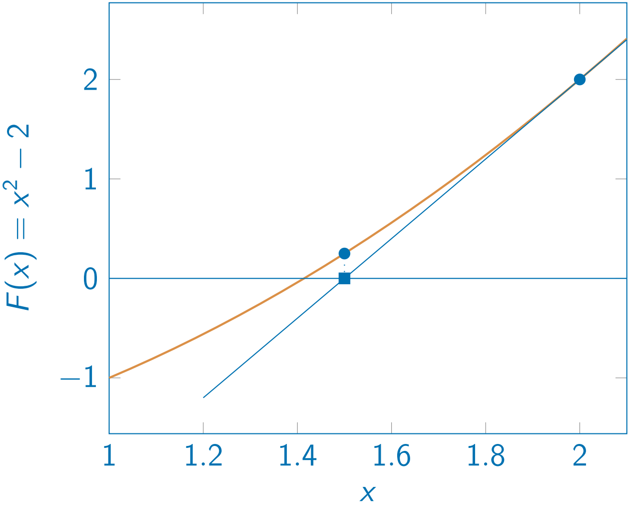
Example



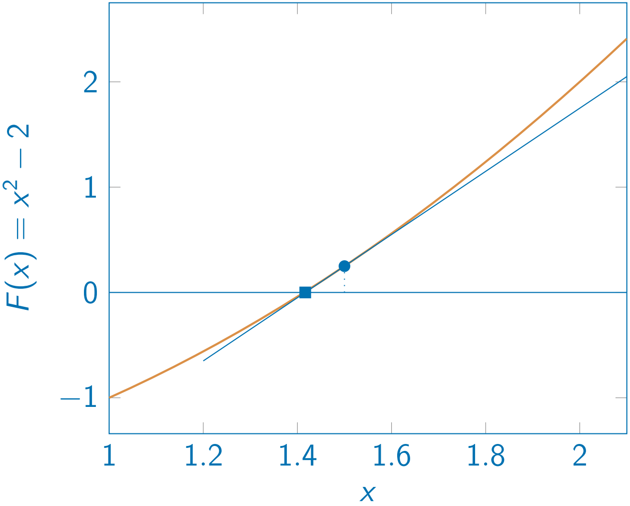
Example



Example



Example



Iterations

k	x_k	$F(x_k)$	$F'(x_k)$
0	+2.00000000E+00	+2.00000000E+00	+4.00000000E+00
1	+1.50000000E+00	+2.50000000E-01	+3.00000000E+00
2	+1.41666667E+00	+6.94444444E-03	+2.83333333E+00
3	+1.41421569E+00	+6.00730488E-06	+2.82843137E+00
4	+1.41421356E+00	+4.51061410E-12	+2.82842712E+00
5	+1.41421356E+00	+4.44089210E-16	+2.82842712E+00

Speed of convergence

- Fast method.
- The precision doubles at each iteration.

Linear model

Taylor's theorem

$$F : \mathbb{R} \to \mathbb{R} \text{ differentiable}$$

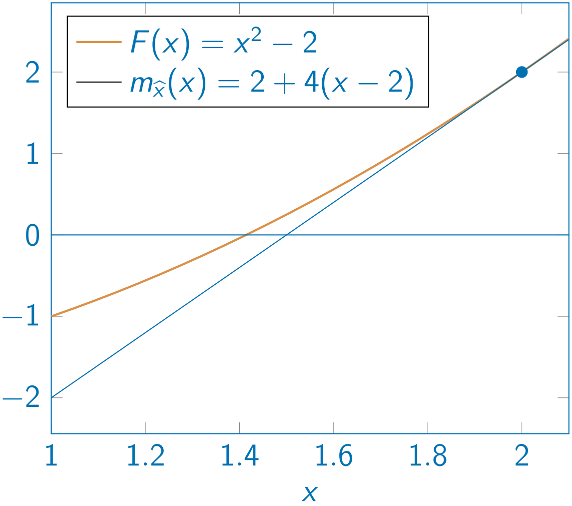
$$m_{\widehat{x}}(x) = F(\widehat{x}) + (x - \widehat{x})F'(\widehat{x})$$

Example

$$F(x) = x^2 - 2F'(x) = 2x$$

$$\widehat{x} = 2 : m_{\widehat{x}}(x) = 2 + 4(x - 2)$$

Example



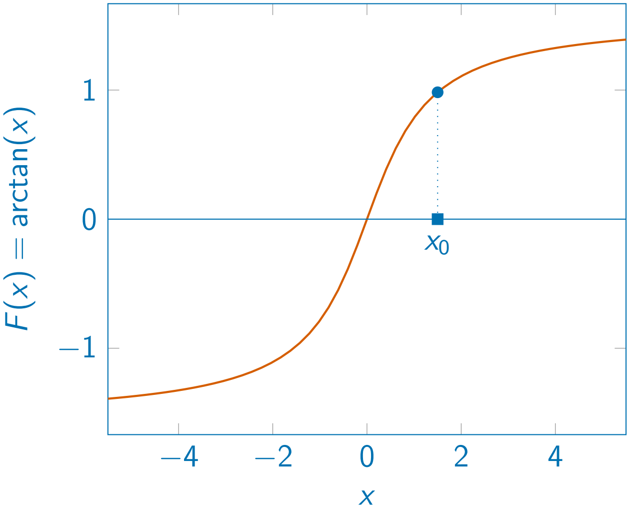
Algorithm

At each iteration, find x_{k+1} that solves

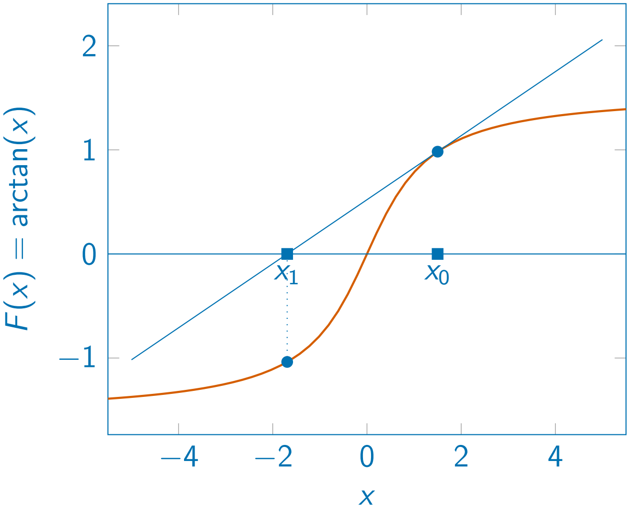
$$F(x_k) + (x - x_k)F'(x_k) = 0$$

$$\begin{aligned}
(x_{k+1} - x_k)F'(x_k) &= -F(x_k) \\
x_{k+1} - x_k &= -F(x_k)/F'(x_k) \\
x_{k+1} &= x_k - F(x_k)/F'(x_k)
\end{aligned}$$

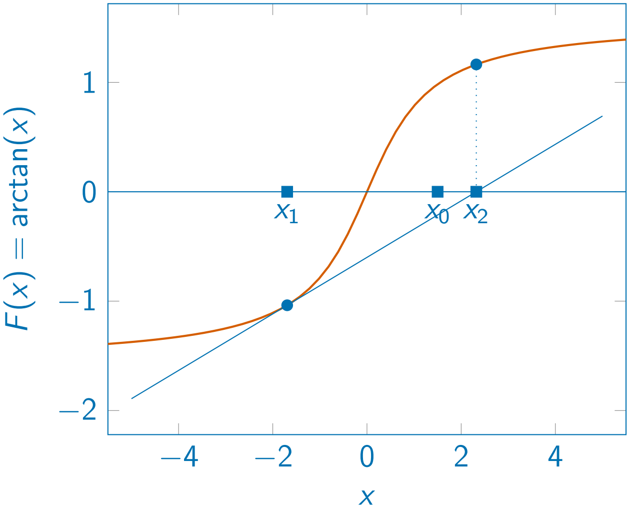
Another example



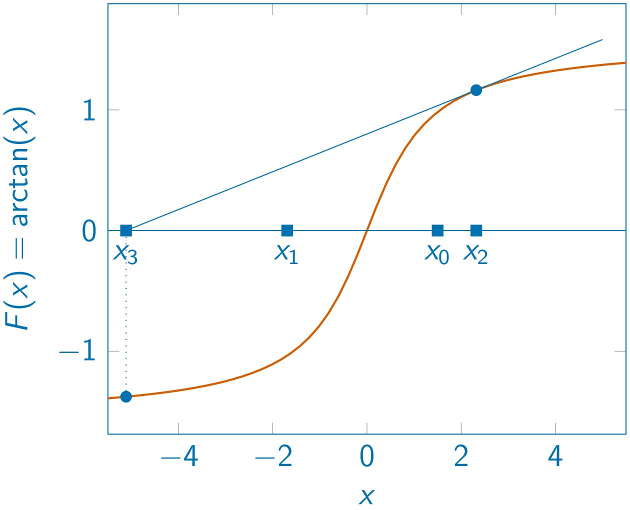
Another example



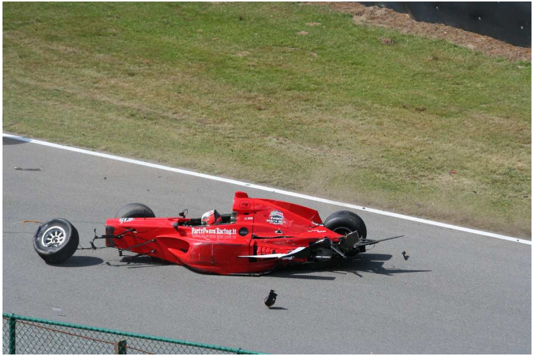
Another example



Another example



Convergence



- The method is fast...
- but does not always converge.
- Several conditions have to be met.

Convergence

Motivation

- We have seen that Newton's method, when it converges, does so fast.
- But it does not always converge.
- We see now in what circumstances it converges, and quantify its speed.
- Then, we generalize the method for systems of equations.

Conditions

F is not too non linear

F' is Lipschitz continuous, with Lipschitz constant M.

$$\exists M > 0 \text{ such that } \forall x, y,\ |F'(x) - F'(y)| \leq M|x - y|.$$

Conditions

F is not too non linear

F' is Lipschitz continuous, with Lipschitz constant M.

$$\exists M > 0 \text{ such that } \forall x, y,\ |F'(x) - F'(y)| \leq M|x - y|.$$

F' is not too close to 0

$$\exists \rho > 0 \text{ such that } |F'(x)| \geq \rho, \forall x.$$

Conditions

F is not too non linear

F' is Lipschitz continuous, with Lipschitz constant M.

$$\exists M > 0 \text{ such that } \forall x, y,\ |F'(x) - F'(y)| \leq M|x - y|.$$

F' is not too close to 0

$$\exists \rho > 0 \text{ such that } |F'(x)| \geq \rho, \forall x.$$

x_0 is not too far from the solution

$$\exists \eta > 0 \text{ such that } |x_0 - x^*| < \eta.$$

Convergence

Theorem 7.7

Consider the sequence of iterates:

$$x_{k+1} = x_k - \frac{F(x_k)}{F'(x_k)}, \qquad k = 0, 1, \dots ,$$

- it is well defined,
- $\lim_{k\to\infty} x_k = x^*$,
- it converges q-quadratically:

$$\left|x_{k+1} - x^*\right| \leq \frac{M}{2\rho} \left|x_k - x^*\right|^2 .$$

Extension to n variables

Problem statement

$$F : \mathbb{R}^n \to \mathbb{R}^n \;\; \text{differentiable}$$

Find $x \in \mathbb{R}^n$ such that

$$F(x) = 0.$$

Reminder

$$F : \mathbb{R}^n \to \mathbb{R}^n$$

Gradient matrix

$$\nabla F(x) = \begin{pmatrix} | & & | \\ \nabla F_1(x) & \cdots & \nabla F_n(x) \\ | & & | \end{pmatrix}$$

$$= \begin{pmatrix} \frac{\partial F_1}{\partial x_1} & \frac{\partial F_2}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_1} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial F_1}{\partial x_n} & \frac{\partial F_2}{\partial x_n} & \cdots & \frac{\partial F_n}{\partial x_n} \end{pmatrix}.$$

Jacobian matrix

$$J(x) = \nabla F(x)^T$$

$$= \begin{pmatrix} \text{———} & \nabla F_1(x)^T & \text{———} \\ & \vdots & \\ \text{———} & \nabla F_n(x)^T & \text{———} \end{pmatrix}.$$

Linear model

$$m_{\widehat{x}}(x) = F(\widehat{x}) + \nabla F\left(\widehat{x}\right)^T (x - \widehat{x}) = F(\widehat{x}) + J(\widehat{x})(x - \widehat{x})$$

Iterations

$$x_{k+1} = x_k + d_k$$

where

$$J(x_k)d_k = -F(x_k).$$

Solving the necessary optimality conditions

Motivation

- The necessary optimality condition stipulates that, at a local optimum, the gradient is zero.
- The first algorithm consists in solving the system of equations $\nabla f(x) = 0$ in order to find a stationary point...
- ... using Newton's method.

Newton's method

Equations

Problem

$$F(x) = 0$$

Algorithm

$$x_{k+1} = x_k + d_k$$

where d_k is solution of

$$J(x_k)d_k = -F(x_k).$$

Optimization

Problem

$$\nabla f(x) = 0$$

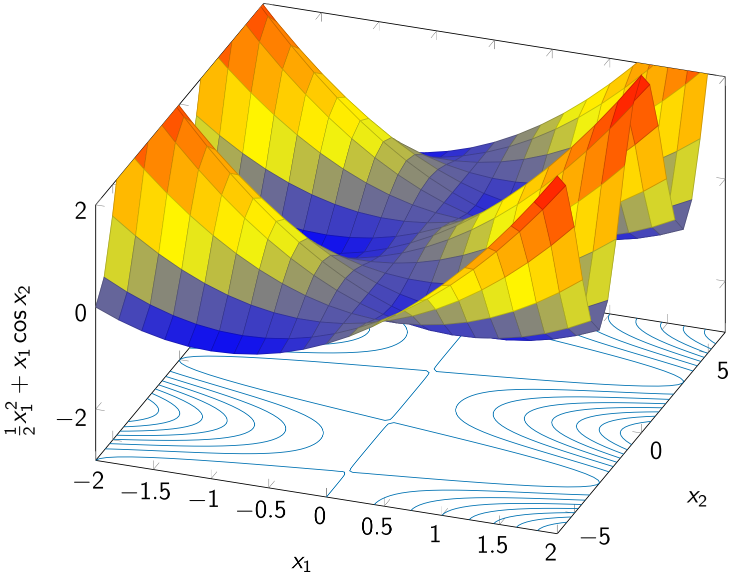
Algorithm

$$x_{k+1} = x_k + d_k$$

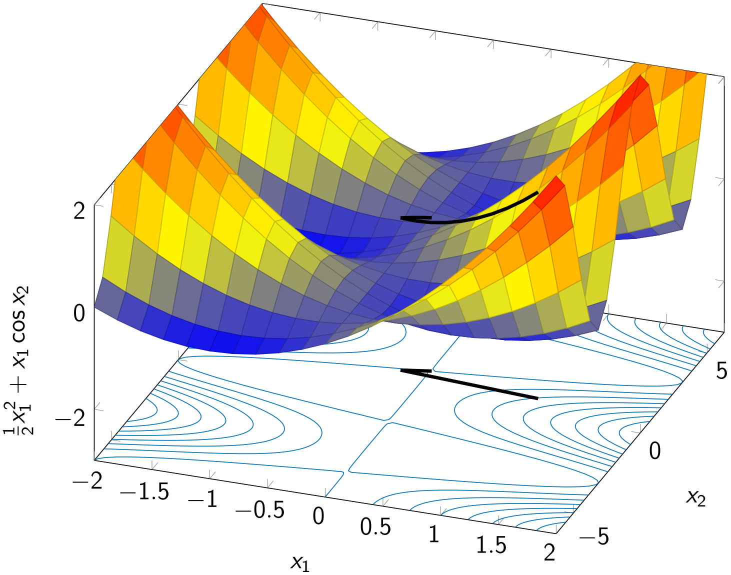
where d_k is solution of

$$\nabla^2 f(x_k)d_k = -\nabla f(x_k).$$

Example



Example: fast convergence



Iterations

k	x_k	$\nabla f(x_k)$	$\lVert \nabla f(x_k) \rVert$	$f(x_k)$
0	1.00000000e+00	1.54030230e+00	1.75516512e+00	1.04030231e+00
	1.00000000e+00	-8.41470984e-01		
1	-2.33845128e-01	-2.87077027e-02	2.30665381e-01	7.53121618e-02
	1.36419220e+00	2.28871986e-01		
2	1.08143752e-02	-3.22524807e-03	1.12840544e-02	-9.33543838e-05
	1.58483641e+00	-1.08133094e-02		
3	-2.13237666e-06	9.22828706e-07	2.32349801e-06	8.79175320e-12
	1.57079327e+00	2.13237666e-06		
4	1.99044272e-17	8.11347449e-17	8.35406072e-17	1.35248527e-25
	1.57079632e+00	-1.99044272e-17		

Solution

Variables

$$x^* = \begin{pmatrix} 0 \\ \pi/2 \end{pmatrix}$$

First derivatives

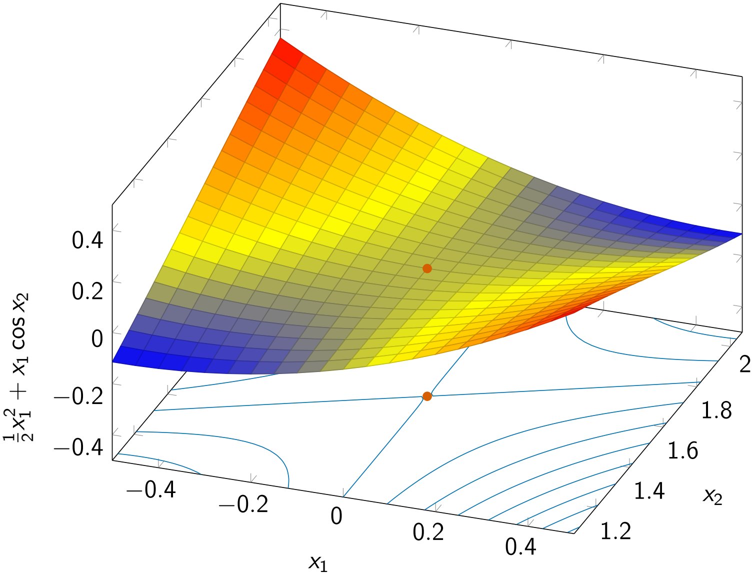
$$\nabla f(x^*) = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Second derivatives

$$\nabla^2 f(x^*) = \begin{pmatrix} 1 & -1 \\ -1 & 0 \end{pmatrix}$$

$$\lambda_1 = -0.61803, \; \lambda_2 = 1.6180$$

Saddle point



Geometric interpretation

Motivation

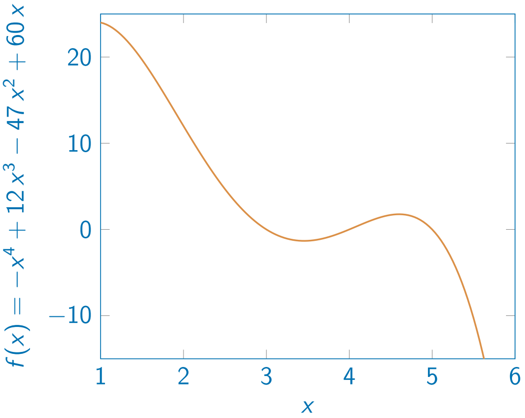
- In the context of equations, Newton's method uses at each iteration a linear model of the function.
- In the context of optimization, it uses a linear model of the gradient of the objective function.
- Equivalently, it uses a quadratic model of the objective function.

Quadratic model

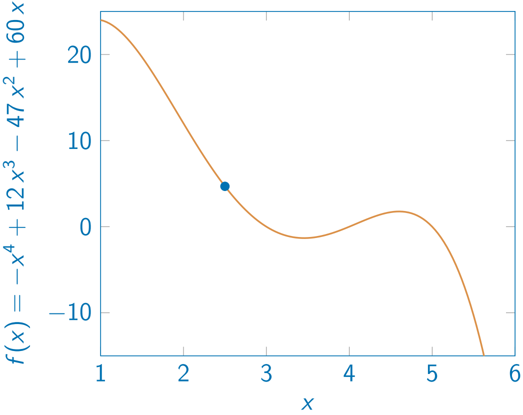
$f : \mathbb{R}^n \to \mathbb{R}$ twice differentiable

$$m_{\widehat{x}}(x) = f(\widehat{x}) + (x - \widehat{x})^T \nabla f(\widehat{x}) + \frac{1}{2}(x - \widehat{x})^T \nabla^2 f(\widehat{x})(x - \widehat{x})$$

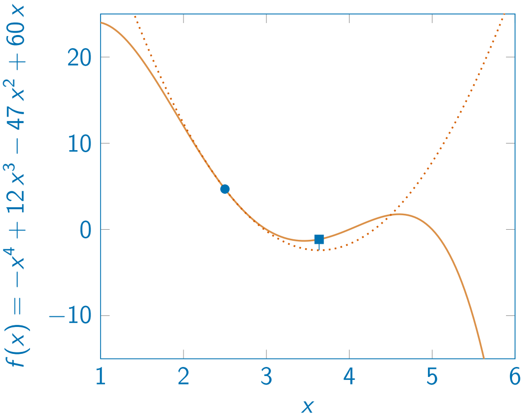
Example



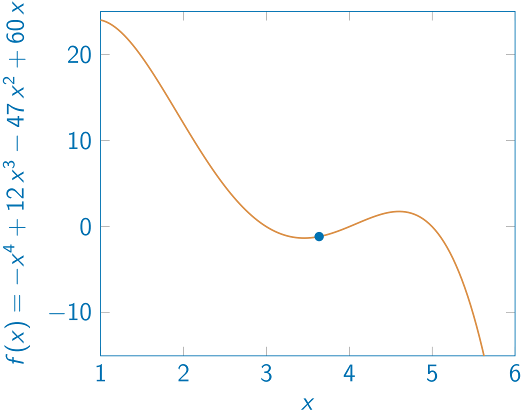
Example: $x_k = 2.5$



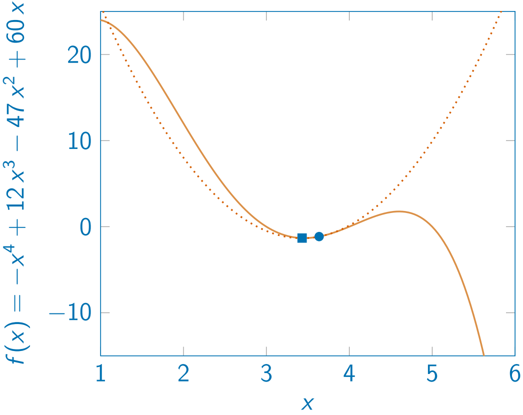
Example: $x_k = 2.5$



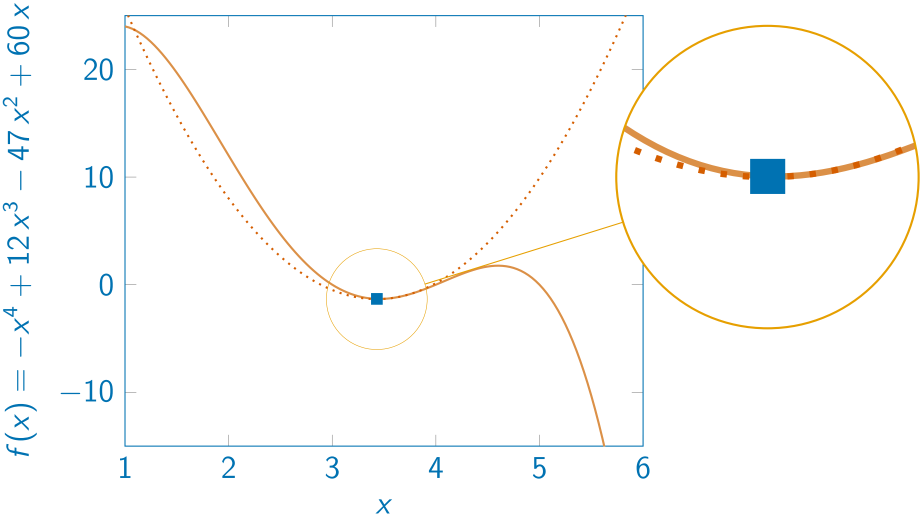
Example: $x_k = 3.636$



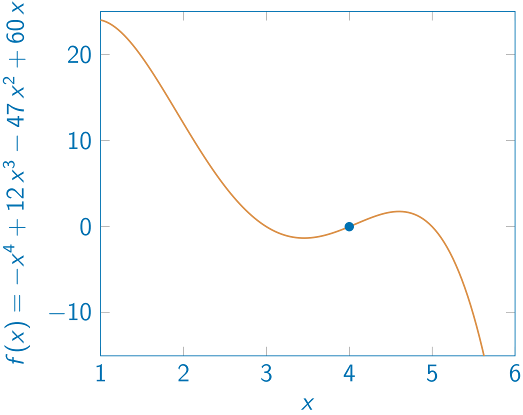
Example: $x_k = 3.636$



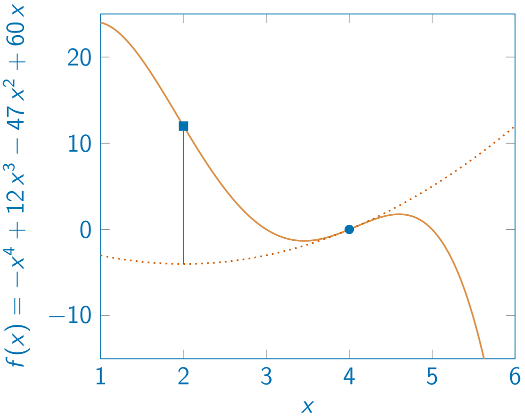
Example: $x_k = 3.636$



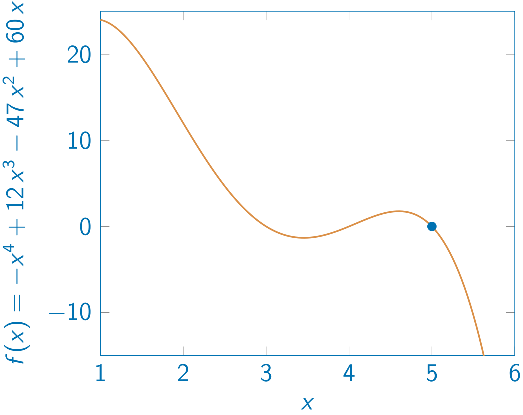
Example: $x_k = 4$



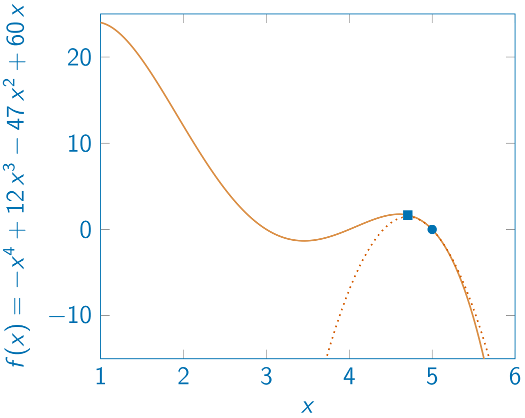
Example: $x_k = 4$



Example: $x_k = 5$



Example: $x_k = 5$



Newton's point

- $f : \mathbb{R}^n \to \mathbb{R}$ twice differentiable
- $x_k \in \mathbb{R}^n$ such that $\nabla^2 f(x_k) > 0$

The Newton's point of f at x_k is

$$x_N = x_k + d_N \, ,$$

where d_N verifies Newton's equations:

$$\nabla^2 f(x_k) d_N = -\nabla f(x_k) \, .$$

Cauchy's point

- $f : \mathbb{R}^n \to \mathbb{R}$ twice differentiable
- $x_k \in \mathbb{R}^n$ such that $\nabla f(x_k)^T \nabla^2 f(x_k) \nabla f(x_k) > 0$

The Cauchy point of f at x_k is

$$x_C = x_k - \alpha_C \nabla f(x_k)\,,$$

where

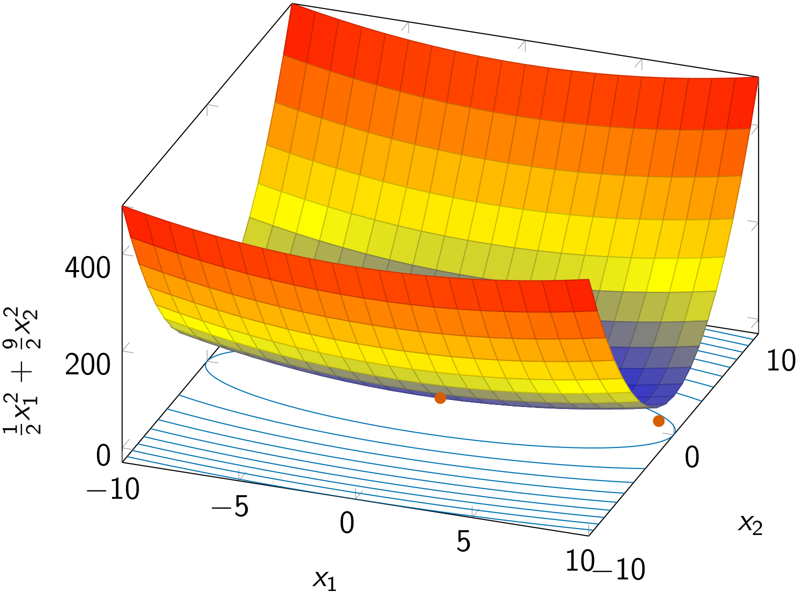
$$\alpha_C = \frac{\nabla f(x_k)^T \nabla f(x_k)}{\nabla f(x_k)^T \nabla^2 f(x_k) \nabla f(x_k)}\,.$$

Preconditioned steepest descent

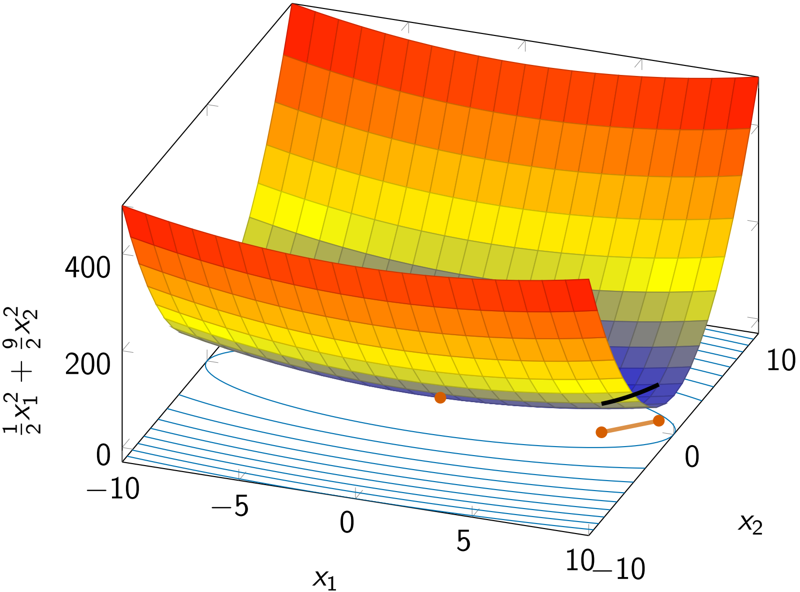
Motivation

- A general idea to minimize a function consists in
  - finding a direction along which the function decreases,
  - follow that direction to find a point with a lower value of the objective function.
- This family of methods is called descent methods.
- A natural direction to follow is the negative gradient, as it is the steepest descent direction.

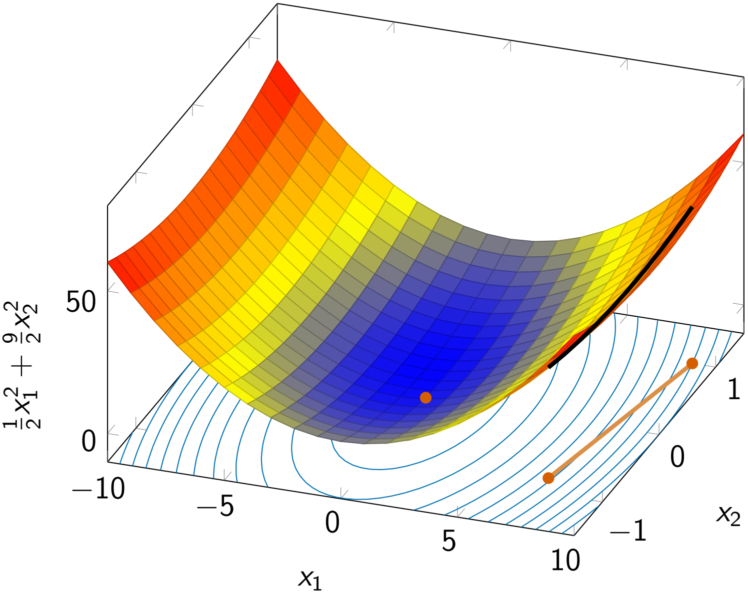
Example



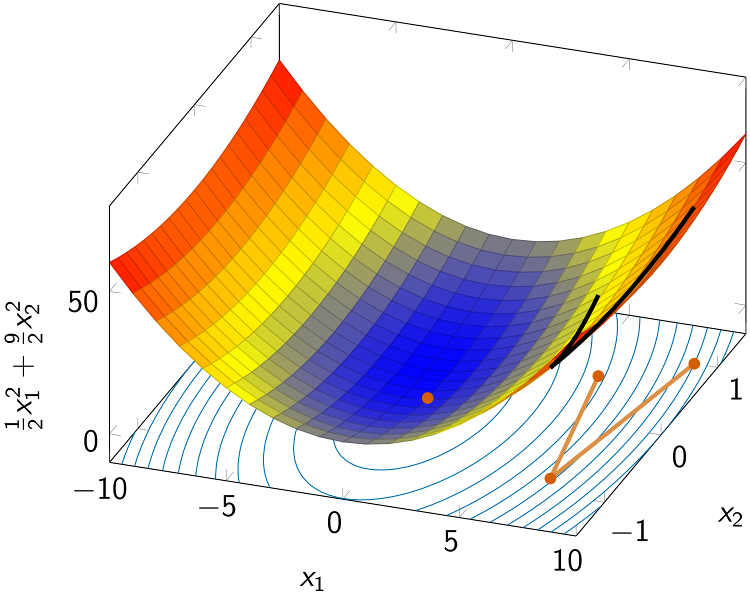
Example



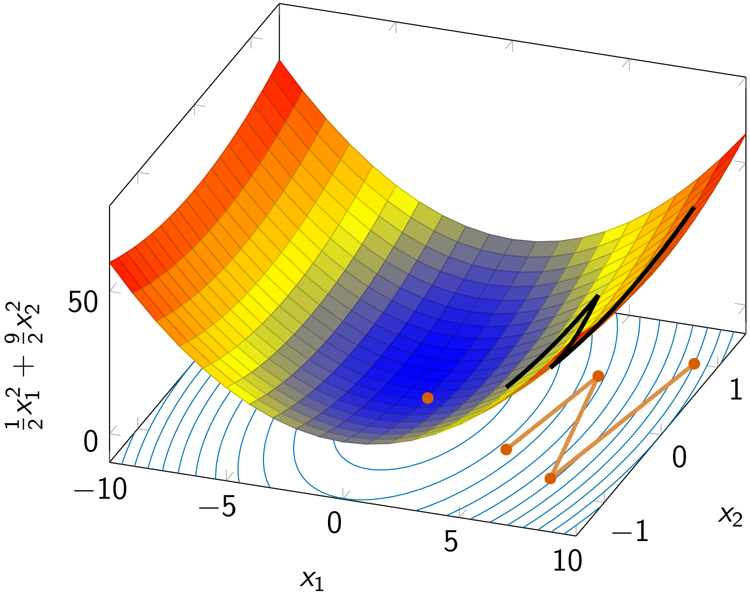
Example



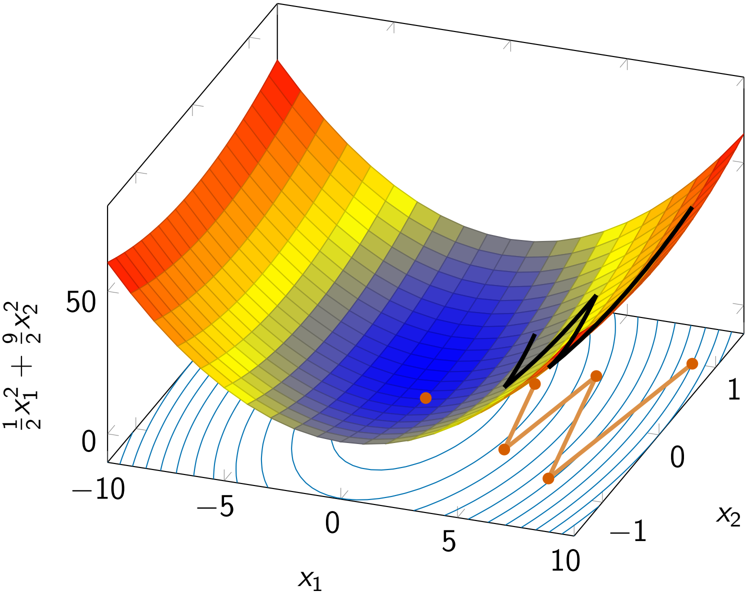
Example



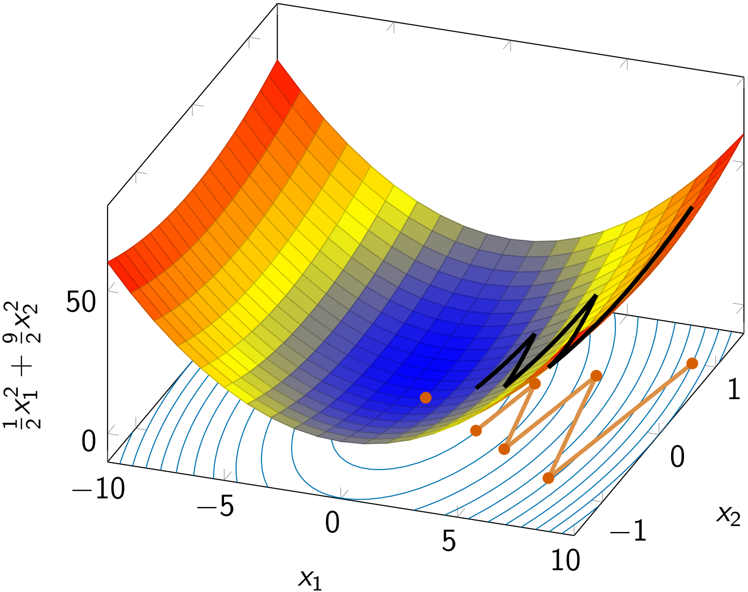
Example



Example



Example



Iterations

k	$(x_k)_1$	$(x_k)_2$	$\nabla f(x_k)_1$	$\nabla f(x_k)_2$	α_k	$f(x_k)$
0	+9.000000E+00	+1.000000E+00	+9.000000E+00	+9.000000E+00	0.2	+4.500000E+01
1	+7.200000E+00	-8.000000E-01	+7.200000E+00	-7.200000E+00	0.2	+2.880000E+01
2	+5.760000E+00	+6.400000E-01	+5.760000E+00	+5.760000E+00	0.2	+1.843200E+01
3	+4.608000E+00	-5.120000E-01	+4.608000E+00	-4.608000E+00	0.2	+1.179648E+01
4	+3.686400E+00	+4.096000E-01	+3.686400E+00	+3.686400E+00	0.2	+7.549747E+00
5	+2.949120E+00	-3.276800E-01	+2.949120E+00	-2.949120E+00	0.2	+4.831838E+00
$\vdots$						
50	+1.284523E-04	+1.427248E-05	+1.284523E-04	+1.284523E-04	0.2	+9.166662E-09
51	+1.027618E-04	-1.141798E-05	+1.027618E-04	-1.027618E-04	0.2	+5.866664E-09
52	+8.220947E-05	+9.134385E-06	+8.220947E-05	+8.220947E-05	0.2	+3.754665E-09
53	+6.576757E-05	-7.307508E-06	+6.576757E-05	-6.576757E-05	0.2	+2.402985E-09
54	+5.261406E-05	+5.846007E-06	+5.261406E-05	+5.261406E-05	0.2	+1.537911E-09
55	+4.209125E-05	-4.676805E-06	+4.209125E-05	-4.209125E-05	0.2	+9.842628E-10

Preconditioning

Original problem

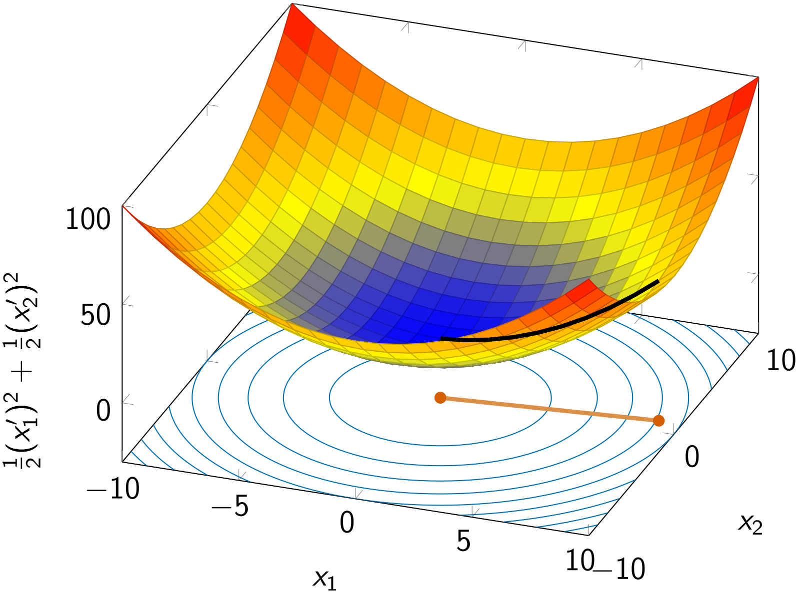
$$\min_{x \in \mathbb{R}^2} f(x) = \frac{1}{2}x_1^2 + \frac{9}{2}x_2^2$$

Change of variables

$$\begin{aligned} x_1' &= x_1 \\ x_2' &= 3x_2. \end{aligned}$$

$$\tilde{f}(x') = \frac{1}{2}\, {x_1'}^2 + \frac{9}{2}\left(\frac{1}{3}\, x_2'\right)^2 = \frac{1}{2}\, {x_1'}^2 + \frac{1}{2}\, {x_2'}^2 .$$

Preconditioning



Preconditioning

- H_k symmetric positive definite.
- $H_k = L_k L_k^T$
- Change of variables:

$$x' = L_k^T x$$

- Steepest descent iteration:

$$x'_{k+1} = x'_k - \alpha_k \nabla \tilde{f}(x'_k)$$

$$\tilde{f}(x') = f(L_k^{-T} x')$$

$$\nabla \tilde{f}(x') = L_k^{-1} \nabla f(L_k^{-T} x')$$

$$x'_{k+1} = x'_k - \alpha_k L_k^{-1} \nabla f(L_k^{-T} x'_k)\,.$$

$$L_k^T x_{k+1} = L_k^T x_k - \alpha_k L_k^{-1} \nabla f(x_k)$$

$$x_{k+1} = x_k - \alpha_k L_k^{-T} L_k^{-1} \nabla f(x_k)$$

$$x_{k+1} = x_k - \alpha_k H_k^{-1} \nabla f(x_k)\,.$$

Preconditioned steepest descent

$$x_{k+1} = x_k + \alpha_k d_k$$

with

$$d_k = -D_k \nabla f(x_k)\,.$$

Preconditioner

Symmetric positive definite matrix. For instance,

$$D_k = H_k^{-1}.$$

Descent direction

$$\nabla f(x_k)^T d_k = -\nabla f(x_k)^T D_k \nabla f(x_k) < 0.$$

Inexact line search

Motivation

- Finding a local optimum along a direction takes a large effort, that may not be justified, as it has to be done at each iteration.
- What about choosing any step that would decrease the function?
- We first show that this does not always work.
- And then we propose something similar, that works.

General idea

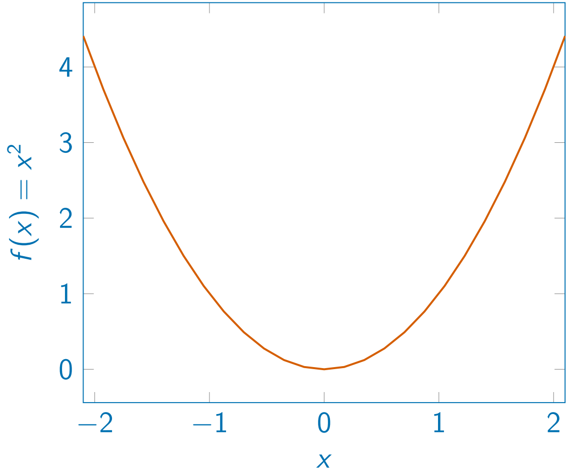
$$f : \mathbb{R}^n \to \mathbb{R},\ x_k \in \mathbb{R}^n,\ d_k \in \mathbb{R}^n,\ \nabla f(x_k)^T d_k < 0.$$

$$x_{k+1} = x_k + \alpha_k d_k,$$

where α_k is such that

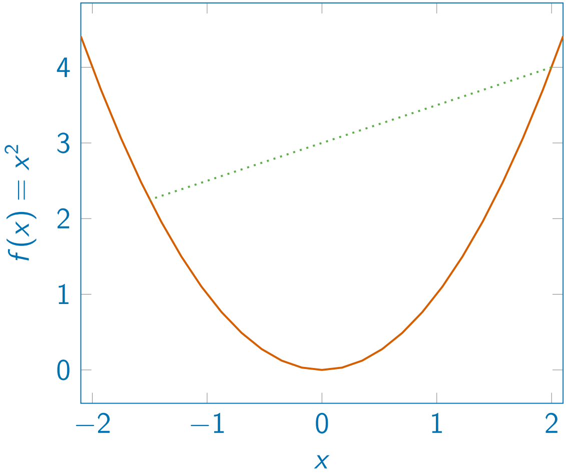
$$f(x_{k+1}) < f(x_k).$$

Example: one dimension



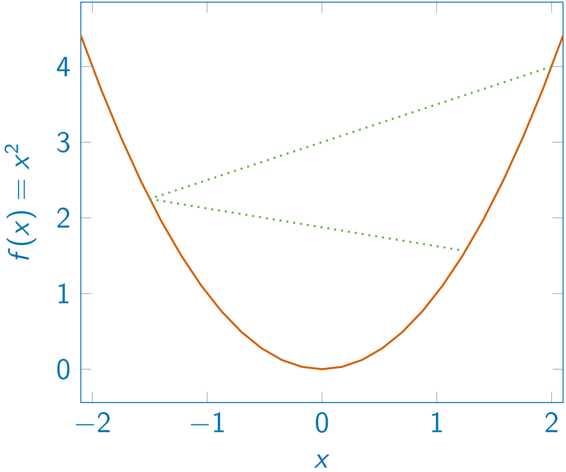
$$f(x) = x^2$$
$$x_0 = 2$$
$$d_k = -\operatorname{sgn}(x_k)$$
$$\alpha_k = 2 + 3(2^{-k-1}) .$$

Example: one dimension



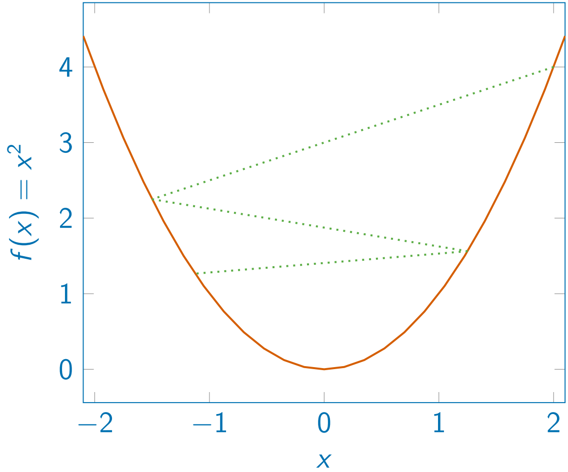
k	x_k	d_k	α_k
0	+2.000000e+00	-1	+3.500000e+00

Example: one dimension



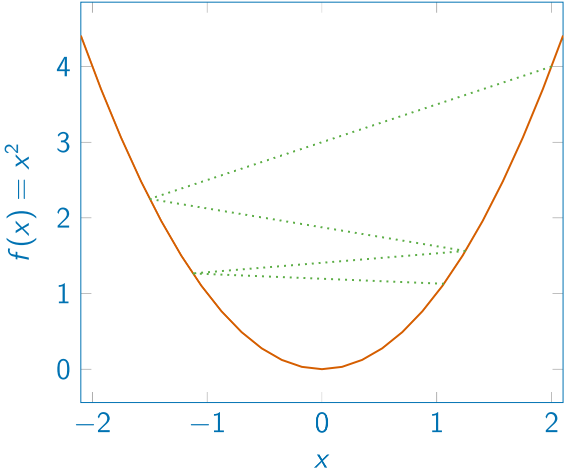
k	x_k	d_k	α_k
0	+2.000000e+00	-1	+3.500000e+00
1	-1.500000e+00	1	+2.750000e+00

Example: one dimension



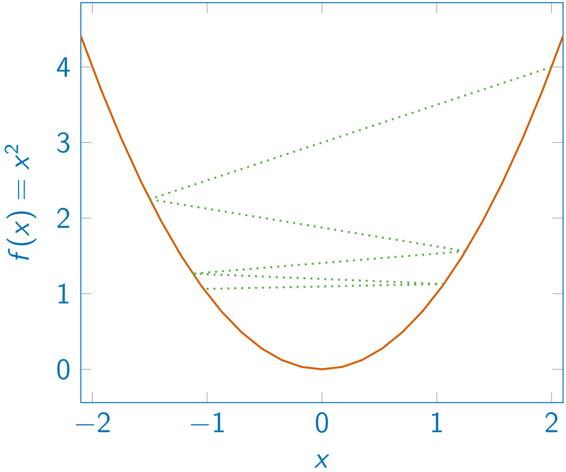
k	x_k	d_k	α_k
0	+2.000000e+00	-1	+3.500000e+00
1	-1.500000e+00	1	+2.750000e+00
2	+1.250000e+00	-1	+2.375000e+00

Example: one dimension



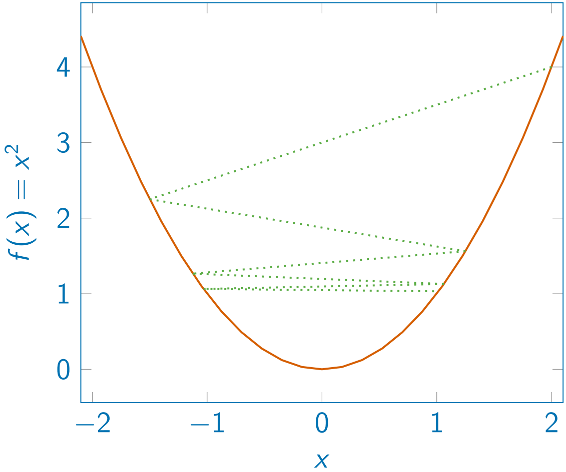
k	x_k	d_k	α_k
0	+2.000000e+00	-1	+3.500000e+00
1	-1.500000e+00	1	+2.750000e+00
2	+1.250000e+00	-1	+2.375000e+00
3	-1.125000e+00	1	+2.187500e+00

Example: one dimension



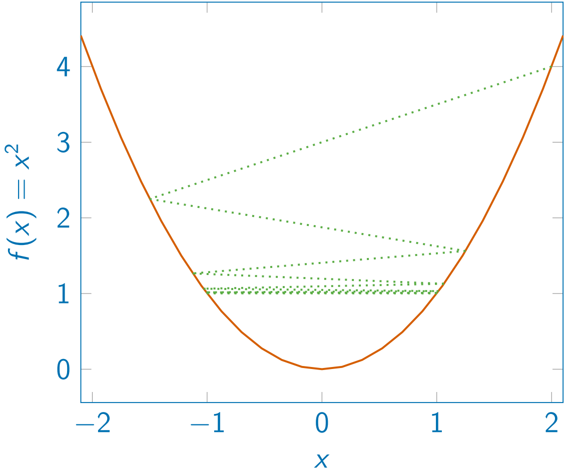
k	x_k	d_k	α_k
0	+2.000000e+00	-1	+3.500000e+00
1	-1.500000e+00	1	+2.750000e+00
2	+1.250000e+00	-1	+2.375000e+00
3	-1.125000e+00	1	+2.187500e+00
4	+1.062500e+00	-1	+2.093750e+00

Example: one dimension



k	x_k	d_k	α_k
0	+2.000000e+00	-1	+3.500000e+00
1	-1.500000e+00	1	+2.750000e+00
2	+1.250000e+00	-1	+2.375000e+00
3	-1.125000e+00	1	+2.187500e+00
4	+1.062500e+00	-1	+2.093750e+00
5	-1.031250e+00	1	+2.046875e+00

Example: one dimension



k	x_k	d_k	α_k
0	+2.000000e+00	-1	+3.500000e+00
1	-1.500000e+00	1	+2.750000e+00
2	+1.250000e+00	-1	+2.375000e+00
3	-1.125000e+00	1	+2.187500e+00
4	+1.062500e+00	-1	+2.093750e+00
5	-1.031250e+00	1	+2.046875e+00
$\vdots$			
46	+1.000000e+00	-1	+2.000000e+00
47	-1.000000e+00	1	+2.000000e+00
48	+1.000000e+00	-1	+2.000000e+00
49	-1.000000e+00	1	+2.000000e+00
50	+1.000000e+00	-1	+2.000000e+00

Example: one dimension

$$x_k = (-1)^k(1 + 2^{-k})$$

$$|x_{k+1}| < |x_k| \text{ therefore } x_{k+1}^2 < x_k^2 \text{ and } f(x_{k+1}) < f(x_k).$$

$$\lim_{k\to\infty} x_k \text{ does not exists}$$

Two subsequences converging to -1 and 1.

Intuition

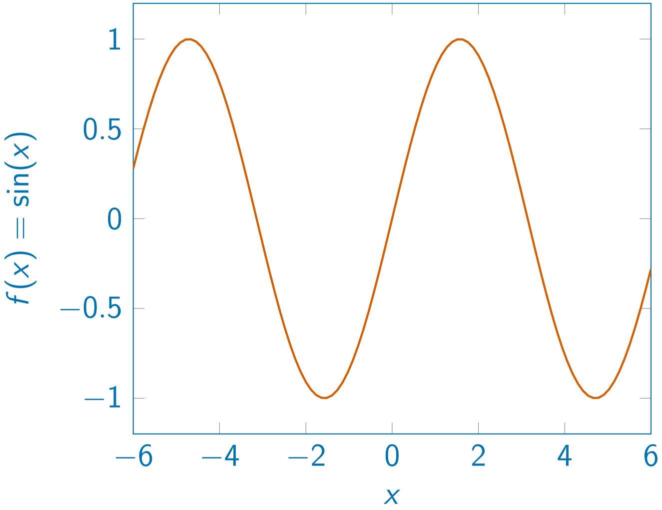
- Descent direction is a local concept.
- It assumes small steps.

Intuition

- Descent direction is a local concept.
- It assumes small steps.
- When steps are too long, Taylor's theorem does not apply anymore.

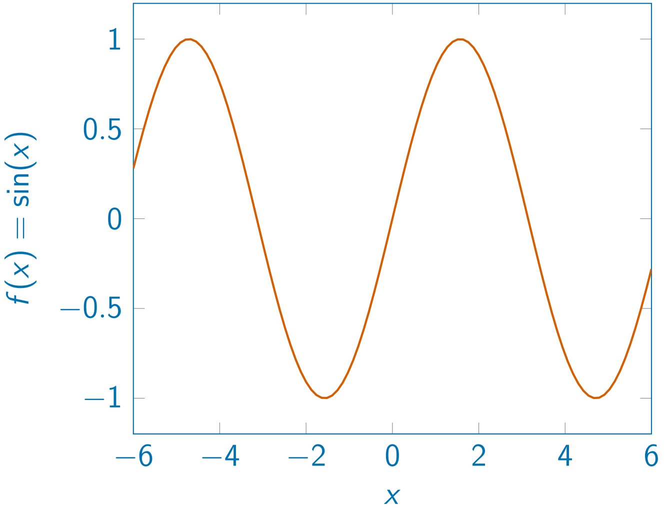
Intuition

- Descent direction is a local concept.
- It assumes small steps.
- When steps are too long, Taylor's theorem does not apply anymore.
- Even ascent directions can decrease the function with long steps.



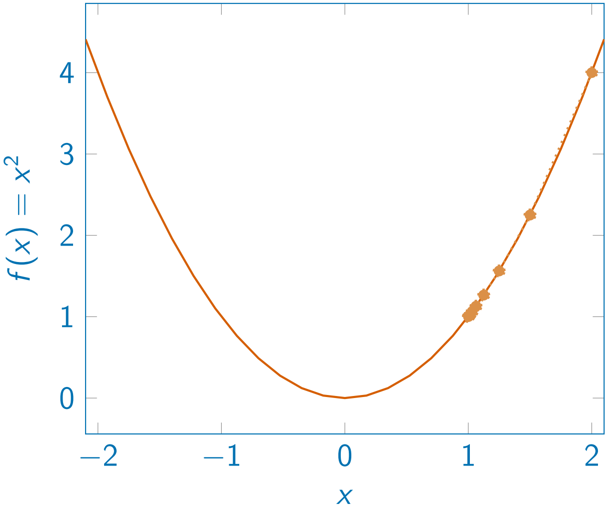
Intuition

- Descent direction is a local concept.
- It assumes small steps.
- When steps are too long, Taylor's theorem does not apply anymore.
- Even ascent directions can decrease the function with long steps.
- Solution: avoid long steps.



Second example

$$f(x) = x^2$$
$$x_0 = 2$$
$$d_k = -1$$
$$\alpha_k = 2^{-k-1}$$
$$x_k = 1 + 2^{-k} > 0$$
$$x_{k+1} < x_k \iff f(x_{k+1}) < f(x_k)$$
$$\lim_{k\to\infty} x_k = 1$$

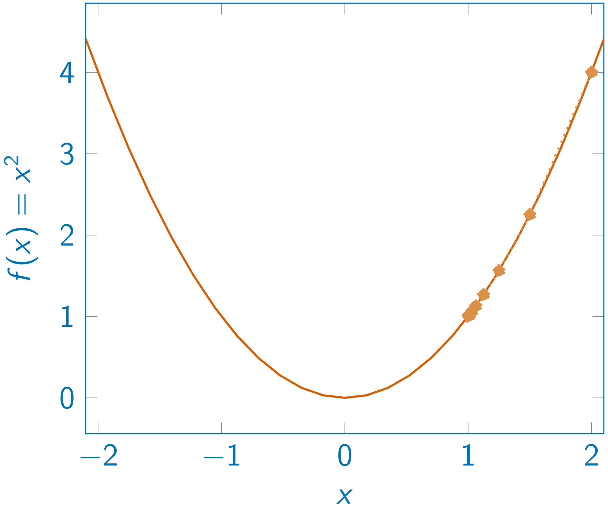


Intuition

- Steps become smaller and smaller:

$$\lim_{k\to\infty} \alpha_k = \lim_{k\to\infty} 2^{-k-1} = 0.$$

- Although $\alpha_k > 0$, almost no progress can be made.
- Solution: avoid short steps.



First Wolfe condition

Motivation

- Our objective is to identify potential steps that can be done along a descent direction.
- We saw that the algorithm may fail to converge if the steps are too long.
- We provide here a characterization of the concept of “being too long”.

First idea

$$f : \mathbb{R}^n \to \mathbb{R},\ x_k \in \mathbb{R}^n,\ d_k \in \mathbb{R}^n,\ \nabla f(x_k)^T d_k < 0.$$

Require a decrease proportional to the step

$$f(x_k) - f(x_k + \alpha_k d_k) \geq \alpha_k \gamma\,,$$

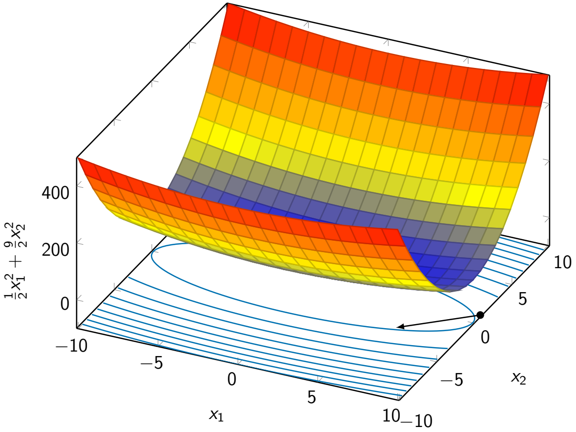
or

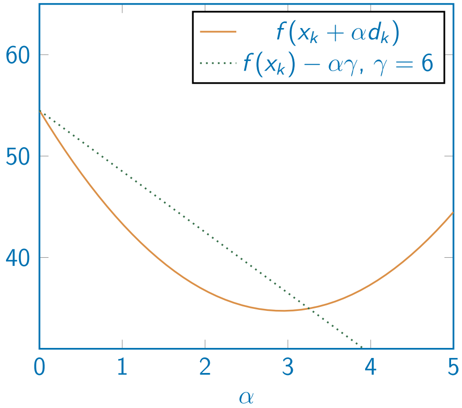
$$f(x_k + \alpha_k d_k) \leq f(x_k) - \alpha_k \gamma\,,$$

with

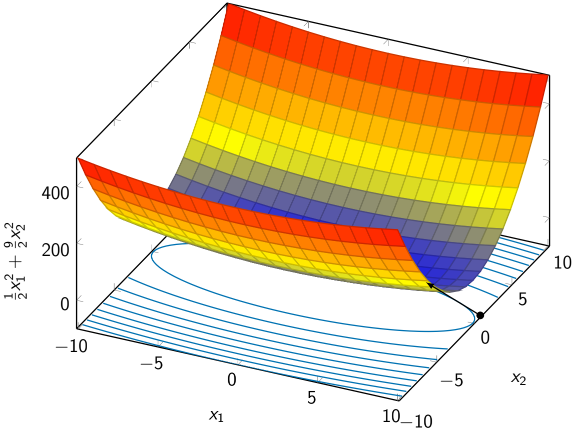
$$\gamma > 0.$$

Decrease proportional to the step

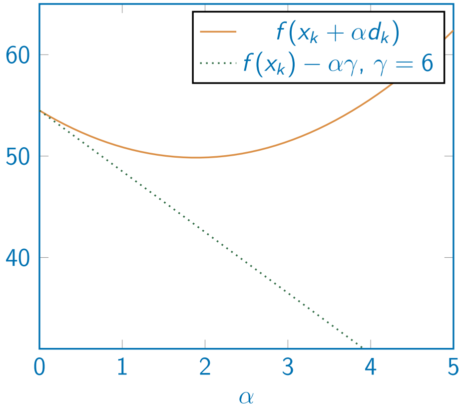




But...



$\frac{1}{2}x_1^2 + \frac{9}{2}x_2^2$
400
200
0
−10
−5
0
5
10
x_1
−10
−5
0
5
10
x_2



$f(x_k + \alpha d_k)$
$f(x_k) - \alpha\gamma,\ \gamma = 6$
60
50
40
0
1
2
3
4
5
α

Second idea

More formally

$$f : \mathbb{R}^n \to \mathbb{R},\ x_k \in \mathbb{R}^n,\ d_k \in \mathbb{R}^n,\ \nabla f(x_k)^T d_k < 0.$$

Choose γ according to the slope:

$$\gamma = -\beta_1 \nabla f(x_k)^T d_k\,, 0 < \beta_1 < 1$$

First Wolfe condition

$$f(x_k + \alpha_k d_k) \leq f(x_k) - \alpha_k \gamma,$$
$$\gamma = -\beta_1 \nabla f(x_k)^T d_k.$$

$$f(x_k + \alpha_k d_k) \leq f(x_k) + \alpha_k \beta_1 \nabla f(x_k)^T d_k.$$

Extreme cases are excluded

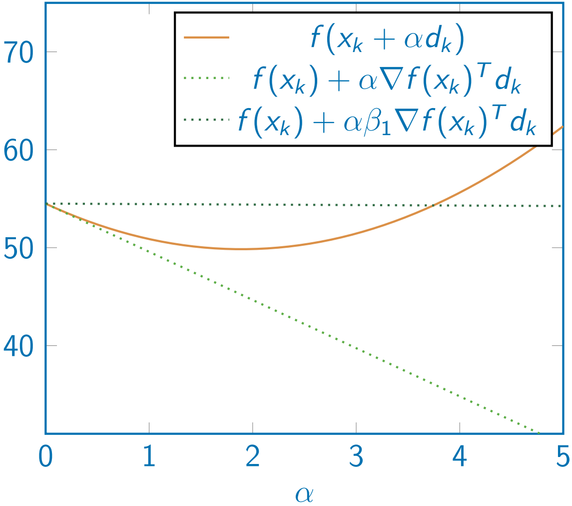
$\beta_1 = 0$

$$f(x_k + \alpha_k d_k) \leq f(x_k).$$

$\beta_1 = 1$

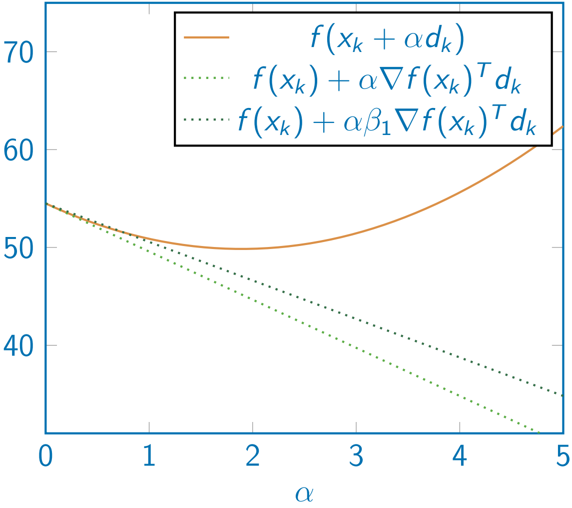
$$f(x_k + \alpha_k d_k) \leq f(x_k) + \alpha_k \nabla f(x_k)^T d_k.$$

$\beta_1 = 0.01$



70
60
50
40
0
1
2
3
4
5
α
$f(x_k + \alpha d_k)$
$f(x_k) + \alpha \nabla f(x_k)^T d_k$
$f(x_k) + \alpha \beta_1 \nabla f(x_k)^T d_k$

$\beta_1 = 0.8$



$f(x_k + \alpha d_k)$
$f(x_k) + \alpha \nabla f(x_k)^T d_k$
$f(x_k) + \alpha \beta_1 \nabla f(x_k)^T d_k$
70
60
50
40
0
1
2
3
4
5
α

Second Wolfe condition

Motivation

- Our objective is to identify valid steps along a descent direction.
- We have first identified a condition that prohibits steps that are too long.
- We are now deriving a condition that prohibits steps that are too short.

Main idea

$$f : \mathbb{R}^n \to \mathbb{R},\ x_k \in \mathbb{R}^n,\ d_k \in \mathbb{R}^n,\ \nabla f(x_k)^T d_k < 0.$$

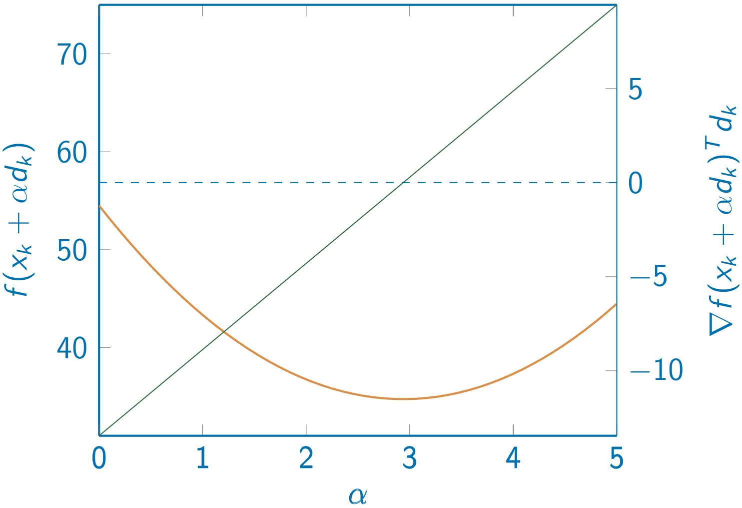
Main idea

$$f : \mathbb{R}^n \to \mathbb{R},\ x_k \in \mathbb{R}^n,\ d_k \in \mathbb{R}^n,\ \nabla f(x_k)^T d_k < 0.$$

Directional derivative along $x_k + \alpha d_k$

- At $\alpha = 0$:
$$\nabla f(x_k)^T d_k < 0.$$
- At α^*, first local minimum:
$$\nabla f(x_k + \alpha^* d_k)^T d_k = 0.$$
- If we reach α^*, the directional derivative increases by $\nabla f(x_k)^T d_k$.

Directional derivative



Second Wolfe condition

Relative reduction of the derivative

$$\frac{\nabla f(x_k + \alpha_k d_k)^T d_k}{\nabla f(x_k)^T d_k} \leq \beta_2$$

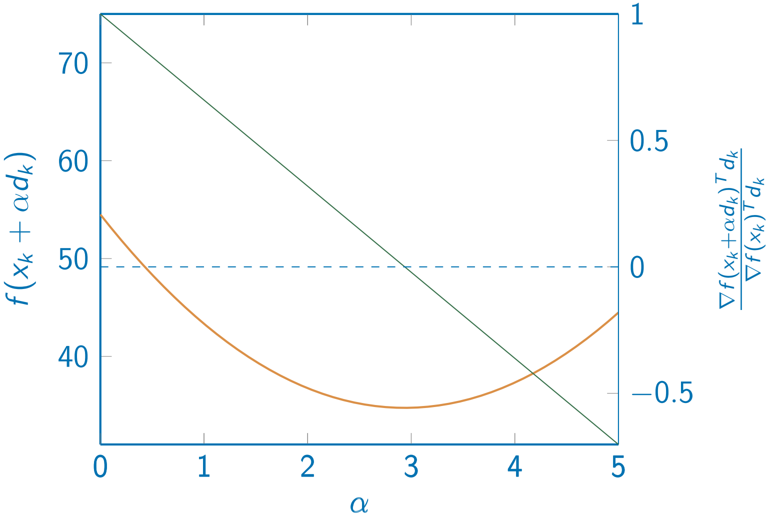
with $0 < \beta_2 < 1$.

Sufficient progress

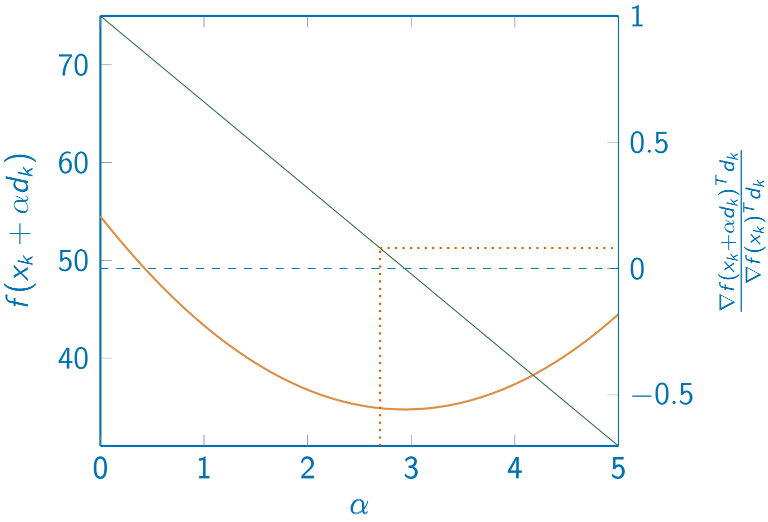
$$\nabla f(x_k + \alpha_k d_k)^T d_k \geq \beta_2 \nabla f(x_k)^T d_k$$

as $\nabla f(x_k)^T d_k < 0$.

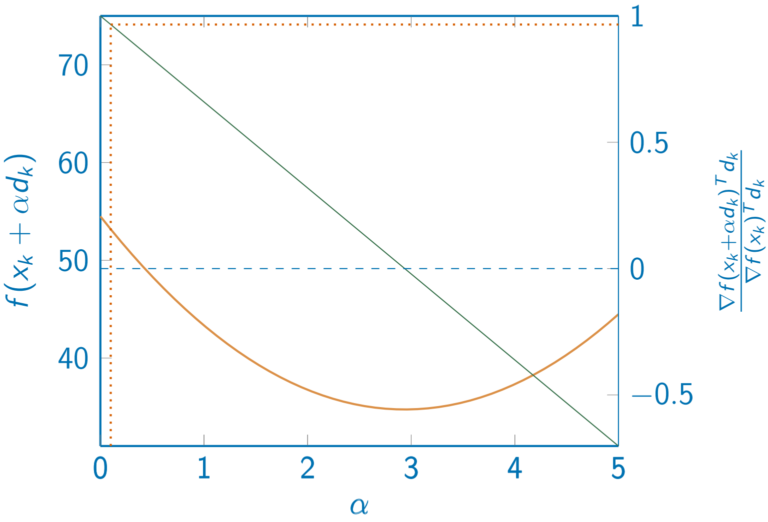
Sufficient progress



β_2 close to 0: larger steps



β_2 close to 1: smaller steps



Validity of the Wolfe conditions

Motivation

- First Wolfe condition forbids steps that are too long.
- Second Wolfe condition forbids steps that are too short.
- How do we guarantee that there exists steps that verify both?

Wolfe conditions

First Wolfe condition

$$f(x_k + \alpha_k d_k) \leq f(x_k) + \alpha_k \beta_1 \nabla f(x_k)^T d_k,$$

$$0 < \beta_1 < 1.$$

Wolfe conditions

First Wolfe condition

$$f(x_k + \alpha_k d_k) \le f(x_k) + \alpha_k \beta_1 \nabla f(x_k)^T d_k,$$

$$0 < \beta_1 < 1.$$

Second Wolfe condition

$$\frac{\nabla f(x_k + \alpha_k d_k)^T d_k}{\nabla f(x_k)^T d_k} \le \beta_2,$$

$$0 < \beta_2 < 1.$$

Compatibility of the Wolfe conditions

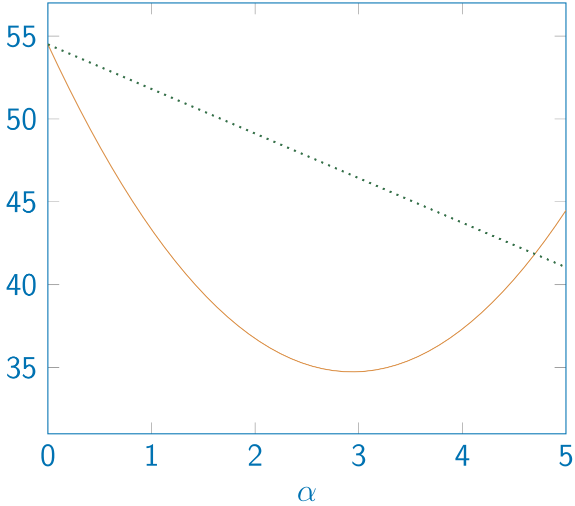
Theorem 11.9

- If f is bounded from below along d_k,
- if $0 < \beta_1 < \beta_2 < 1$,
- there exists $\alpha > 0$ verifying both conditions.

First Wolfe condition

- As d_k is a descent direction,
- $\exists \eta$ such that Wolfe 1 is verified for $\alpha_k \leq \eta$.
- See Theorem 2.11.

Validity of the conditions



Ideas of the proof

See Figure 11.15 p. 273

- Start with the line: $f(x_k) + \alpha\beta_1 \nabla f(x_k)^T d_k$
- α_1: intersection between the line and the function.
- α_1 exists because the function is bounded from below.
- All $\alpha \leq \alpha_1$ verify Wolfe 1.
- α_2: mean value theorem (tangent with same slope). Verifies Wolfe 1.
- As the slope is defined by β_1, we have

  $$\beta_1 \nabla f(x_k)^T d_k = \nabla f(x_k + \alpha_2 d_k)^T d_k$$

  so that

  $$\beta_1 = \frac{\nabla f(x_k + \alpha_2 d_k)^T d_k}{\nabla f(x_k)^T d_k}$$

- If $\beta_2 > \beta_1$, α_2 verifies Wolfe 2.

Linesearch algorithm

Initialize: $i \leftarrow 0$, $\alpha_\ell \leftarrow 0$, $\alpha_r \leftarrow +\infty$.

repeat

  if α_i *violates Wolfe 1* **then** the step is too long

    $\alpha_r \leftarrow \alpha_i$,

    $\alpha_{i+1} \leftarrow \dfrac{\alpha_\ell + \alpha_r}{2}$.

  if α_i *does not violate Wolfe 1 but violates Wolfe 2* **then** the step is too short

    $\alpha_\ell \leftarrow \alpha_i$.

    if $\alpha_r < +\infty$, **then**

      $\alpha_{i+1} \leftarrow \dfrac{\alpha_\ell + \alpha_r}{2}$

    else

      $\alpha_{i+1} \leftarrow 10\alpha_i$.

  $i \leftarrow i + 1$.

until α_i *satisfies Wolfe 1 and 2*

$\alpha^* \leftarrow \alpha_i$.

Finiteness of the line search algorithm

Motivation

- The linesearch algorithm is quite simple:
  1. Start with a candidate step.
  2. If it is too long, make it shorter.
  3. If it is too short, make it longer.
- We now need to verify that this process finishes in a finite number of steps.

Wolfe conditions

First Wolfe condition

$$f(x_k + \alpha_k d_k) \leq f(x_k) + \alpha_k \beta_1 \nabla f(x_k)^T d_k.$$

Second Wolfe condition

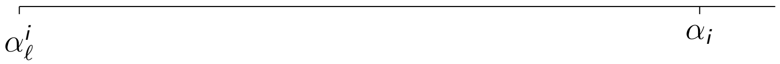
$$\nabla f(x_k + \alpha_k d_k)^T d_k \geq \beta_2 \nabla f(x_k)^T d_k.$$

Parameters

$$0 < \beta_1 < \beta_2 < 1.$$

Line search algorithm

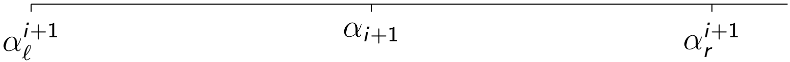
Initialization



Line search algorithm

Initialization

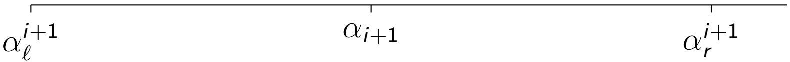
Wolfe 1 violated: step too long



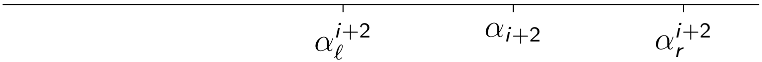
Line search algorithm

Initialization

Wolfe 1 violated: step too long



Wolfe 2 violated: step too short



Line search algorithm

Special case

- If the step is too short (Wolfe 2 violated), and
- $\alpha_r = \infty$,
- then increase the step by an arbitrary factor $\lambda > 1$:

$$\alpha_{i+1} = \lambda \alpha_i .$$

Properties

α_ℓ^i
always verifies Wolfe 1 : it is updated only when Wolfe 1 is verified and Wolfe 2 is violated.

Properties

α^i_ℓ
always verifies Wolfe 1 : it is updated only when Wolfe 1 is verified and Wolfe 2 is violated.

α^i_ℓ
always violates Wolfe 2 for the same reason.

Properties

α_ℓ^i
always verifies Wolfe 1 : it is updated only when Wolfe 1 is verified and Wolfe 2 is violated.

α_ℓ^i
always violates Wolfe 2 for the same reason.

α_r^i
always violates Wolfe 1 : starts from ∞ and updated when Wolfe 1 is violated.

Finiteness of α_i

Arguments

- Suppose, by contradiction, that $\alpha_i \to \infty$.
- It means that α_r^i is always $+\infty$.
- As α_r^i would be updated when Wolfe 1 is violated, it means that Wolfe 1 is never violated.
- Impossible, as the function is bounded from below.

Finiteness of α_i

Arguments

- Suppose, by contradiction, that $\alpha_i \to \infty$.
- It means that α_r^i is always $+\infty$.
- As α_r^i would be updated when Wolfe 1 is violated, it means that Wolfe 1 is never violated.
- Impossible, as the function is bounded from below.

Consider only iterations when $\alpha_r^i < \infty$.

$$\alpha_{i+1} = \frac{\alpha_\ell^i + \alpha_r^i}{2}.$$

Finite number of iterations

Theorem 11.10

- Suppose, by contradiction, that there is an infinite number of iterations.
- Therefore,
$$\lim_{i\to\infty} \alpha_r^i - \alpha_\ell^i = 0.$$
- Consequently,
$$\alpha^* = \lim_{i\to\infty} \alpha_r^i = \lim_{i\to\infty} \alpha_\ell^i = \lim_{i\to\infty} \alpha_i.$$

Finite number of iterations

Theorem 11.10

- Wolfe 1 is verified for all α_ℓ^i. At the limit:

$$f(x_k + \alpha^* d_k) \leq f(x_k) + \alpha^* \beta_1 \nabla f(x_k)^T d_k.$$

- Wolfe 1 is violated by all α_r^i. Therefore $\alpha^* \neq \alpha_r^i$.

$$f(x_k + \alpha_r^i d_k) > f(x_k) + \alpha_r^i \beta_1 \nabla f(x_k)^T d_k.$$

At the limit

$$f(x_k + \alpha^* d_k) \geq f(x_k) + \alpha^* \beta_1 \nabla f(x_k)^T d_k.$$

Finite number of iterations

Theorem 11.10

$$f(x_k + \alpha^* d_k) = f(x_k) + \alpha^* \beta_1 \nabla f(x_k)^T d_k.$$

- Wolfe 1 is violated by all α_r^i. Therefore $\alpha_r^i - \alpha^* > 0$.

$$f(x_k + \alpha_r^i d_k) > f(x_k) + \alpha_r^i \beta_1 \nabla f(x_k)^T d_k.$$

$$f(x_k + \alpha_r^i d_k) > f(x_k + \alpha^* d_k) - \alpha^* \beta_1 \nabla f(x_k)^T d_k + \alpha_r^i \beta_1 \nabla f(x_k)^T d_k.$$

$$f(x_k + \alpha_r^i d_k) > f(x_k + \alpha^* d_k) + (\alpha_r^i - \alpha^*) \beta_1 \nabla f(x_k)^T d_k.$$

$$\frac{f(x_k + \alpha_r^i d_k) - f(x_k + \alpha^* d_k)}{\alpha_r^i - \alpha^*} > \beta_1 \nabla f(x_k)^T d_k.$$

Finite number of iterations

$$\frac{f(x_k + \alpha_r^i d_k) - f(x_k + \alpha^* d_k)}{\alpha_r^i - \alpha^*} > \beta_1 \nabla f(x_k)^T d_k.$$

$$i \to \infty: \ \nabla f(x_k + \alpha^* d_k)^T d_k \geq \beta_1 \nabla f(x_k)^T d_k.$$

As $\beta_2 > \beta_1$ and $\nabla f(x_k)^T d_k < 0$, we have

$$\nabla f(x_k + \alpha^* d_k)^T d_k > \beta_2 \nabla f(x_k)^T d_k.$$

Finite number of iterations

Current result

$$\nabla f(x_k + \alpha^* d_k)^T d_k > \beta_2 \nabla f(x_k)^T d_k.$$

α_ℓ^i violates Wolfe 2

$$\nabla f(x_k + \alpha_\ell^i d_k)^T d_k < \beta_2 \nabla f(x_k)^T d_k, \ \forall i.$$

At the limit

$$\nabla f(x_k + \alpha^* d_k)^T d_k \leq \beta_2 \nabla f(x_k)^T d_k.$$

Contradiction

Newton method with line search

Motivation

- We now go back to Newton's method.
- We have seen that, when it works, it converges fast.
- But it does not always work.
- Let's combine it with the descent methods framework to obtain an efficient algorithm.

Comparison

Descent method

$$x_{k+1} = x_k - \alpha_k D_k \nabla f(x_k).$$

Newton's method

$$x_{k+1} = x_k - \nabla^2 f(x_k)^{-1} \nabla f(x_k).$$

Comparison

Descent method

$$x_{k+1} = x_k - \alpha_k D_k \nabla f(x_k).$$

Newton's method

$$x_{k+1} = x_k - \nabla^2 f(x_k)^{-1} \nabla f(x_k).$$

Newton's method is a descent method

- if $\alpha_k = 1$,
- if $D_k = \nabla^2 f(x_k)^{-1}$ is positive definite.

Modifications

If $\nabla^2 f(x_k)^{-1}$ not positive definite

Choose another D_k:

- $D_k = I$: bad idea. Slow convergence of steepest descent.
- D_k diagonal:

$$D_k(i,i) = \max\left(\varepsilon, \frac{\partial f^2}{\partial x_i^2}(x_k)\right)^{-1},$$

  with $\varepsilon > 0$.
- Inflating $\nabla^2 f(x_k)$:

$$D_k = (\nabla^2 f(x_k) + \tau I)^{-1},$$

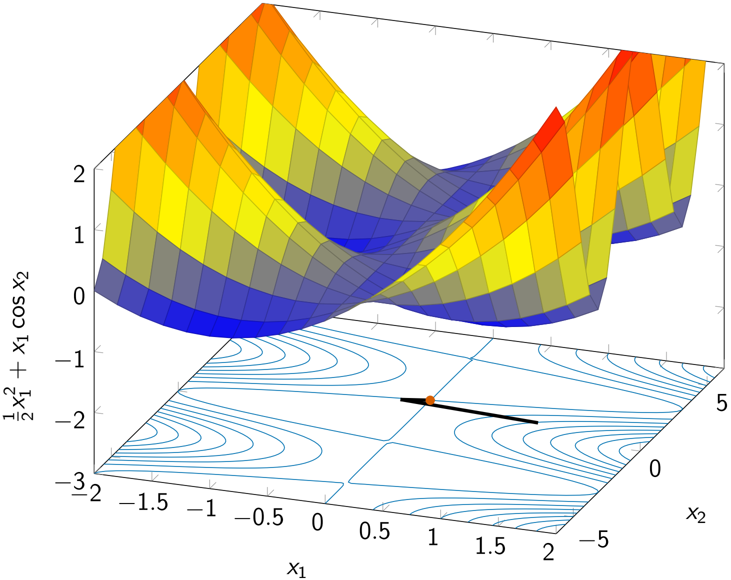
  where τ is calculated such that D_k is positive definite.

Modifications

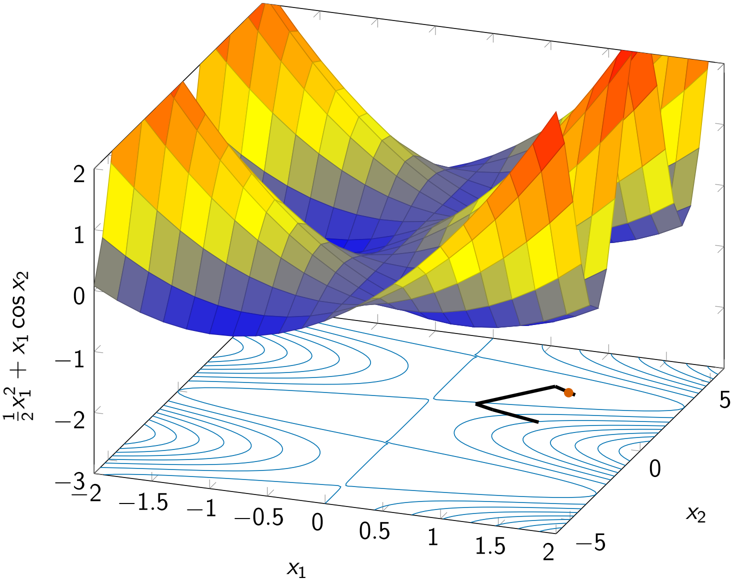
If $\alpha_k = 1$ is not consistent with Wolfe conditions

Apply the line search algorithm.

Example: Newton's method



Example: Modified Newton's method



Iterations

k	$f(x_k)$	$\|\nabla f(x_k)\|$	α_k	τ
0	1.04030231e+00	1.75516512e+00		
1	2.34942031e-01	8.88574897e-01	1	1.64562250e+00
2	4.21849003e-02	4.80063696e-01	1	1.72091923e+00
3	-4.52738278e-01	2.67168927e-01	3	8.64490594e-01
4	-4.93913638e-01	1.14762780e-01	1	0.00000000e+00
5	-4.99982955e-01	5.85174623e-03	1	0.00000000e+00
6	-5.00000000e-01	1.94633135e-05	1	0.00000000e+00
7	-5.00000000e-01	2.18521663e-10	1	0.00000000e+00
8	-5.00000000e-01	1.22460635e-16	1	0.00000000e+00

Fast and reliable

Summary

- Optimality conditions: solve a system of nonlinear equations.
- Newton's method for optimization is fast, but unreliable.
- Other method: preconditioned steepest descent.
- Descent direction: preconditioned gradient.
- Step: Wolfe conditions.
- Newton method with line search.