Function Minimization (I)

Marcin Chrzaszcz, Danny van Dyk
mchrzasz@cern.ch, danny.van.dyk@gmail.com

Numerical Methods,
21. November, 2016

Plan for today

- What types of function do we usually need to minimize?
  - general non-linear problem
  - least-squares

- What information can we use?
  - the target function
  - its first derivatives
  - its second derivatives

General Minimization Problem

The general problem can be posited as follows: There exists a function

$$f(a_1, \ldots, a_N)$$

with real-valued parameters $\{a_n\}$.

We intend to find

- one,
- any, or
- all

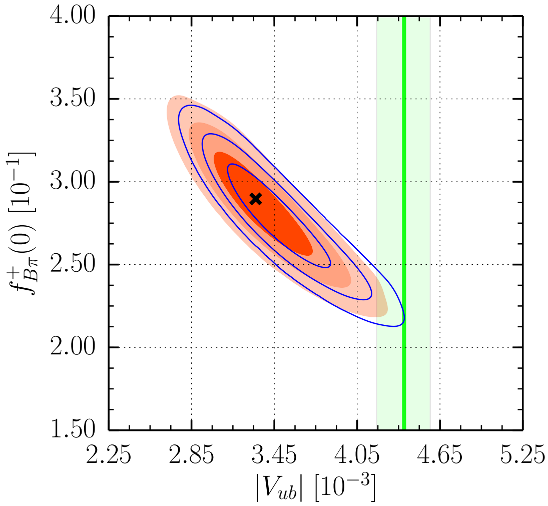
of its minima on a support A,

$$A = \{(a_1, \ldots, a_N) \,|\, g(a_1, \ldots, a_N) > 0\}\,.$$

In order to find maxima, flip the function around: $f \to -f$.

Example

Posterior probability density function with non-gaussian likelihood.



- shallow gradients in one direction
- step gradients in the other
- non-symmetric shape
- slight "banana" qualities (slightly bent in along one axis)

taken from 1409.7186

Limitations and Monte-Carlo Methods

In general we have no further information on f, i.e., we do not *analytically* know any of its derivatives.

A very popular way to explore f is by using Monte Carlo methods, e.g. plain random walks, Markov chain methods, or genetic algorithms. Most of these methds are very good at delineating *local environments* around some/most/all(?) of the modes.

As always, analytic knowledge of the problem will help. For example, symmetry relations among the parameters or (a)periodic boundary conditions should be exploited if possible.

The specific Monte-Carlo methods are beyond the scope of these lectures.

Simplex Method (aka Nelder-Mead Method)

We will now discuss the work-horse of minimization algorithms, the Simplex method. While it has very good convergence properties, but a rather slow convergence rate.

The basic idea is as follows:

0 A set of parameter points must be provided as an initial value. If all else fails, random points can be used.

Simplex Method (aka Nelder-Mead Method)

We will now discuss the work-horse of minimization algorithms, the Simplex method. While it has very good convergence properties, but a rather slow convergence rate.

The basic idea is as follows:

0 A set of parameter points must be provided as an initial value. If all else fails, random points can be used.

1 Order the points by their respetive size of f

Simplex Method (aka Nelder-Mead Method)

We will now discuss the work-horse of minimization algorithms, the Simplex method. While it has very good convergence properties, but a rather slow convergence rate.

The basic idea is as follows:

0 A set of parameter points must be provided as an initial value. If all else fails, random points can be used.

1 Order the points by their respetive size of f

2 Compute the midpoint $\vec{a}_{\text{mid}}$ of all points except the worst, and reflect the worst point $\vec{a}_{N+1}$ at the midpoint yielding $\vec{a}_{\text{ref}}$.

Simplex Method (aka Nelder-Mead Method)

We will now discuss the work-horse of minimization algorithms, the Simplex method. While it has very good convergence properties, but a rather slow convergence rate.

The basic idea is as follows:

0. A set of parameter points must be provided as an initial value. If all else fails, random points can be used.
1. Order the points by their respetive size of f
2. Compute the midpoint $\vec{a}_{\text{mid}}$ of all points except the worst, and reflect the worst point $\vec{a}_{N+1}$ at the midpoint yielding $\vec{a}_{\text{ref}}$.
   a. If $f(\vec{a}_{\text{ref}})$ fulfills some minimization criteria, replace one of the existing simplex points with $\vec{a}_{\text{ref}}$. Continue at step 1.

Simplex Method (aka Nelder-Mead Method)

We will now discuss the work-horse of minimization algorithms, the Simplex method. While it has very good convergence properties, but a rather slow convergence rate.

The basic idea is as follows:

0 A set of parameter points must be provided as an initial value. If all else fails, random points can be used.

1 Order the points by their respetive size of f

2 Compute the midpoint $\vec{a}_{\text{mid}}$ of all points except the worst, and reflect the worst point $\vec{a}_{N+1}$ at the midpoint yielding $\vec{a}_{\text{ref}}$.

   a If $f(\vec{a}_{\text{ref}})$ fulfills some minimization criteria, replace one of the existing simplex points with $\vec{a}_{\text{ref}}$. Continue at step 1.

   b Otherwise compute $\vec{a}_{\text{contr}}$ as a linear combination of the worst point of the simplex $\vec{a}_N$, and $\vec{a}_{\text{ref}}$. If $\vec{a}_{\text{contr}}$ is better than the worst point, replace the worst point. Continue at step 1.

Simplex Method (aka Nelder-Mead Method)

We will now discuss the work-horse of minimization algorithms, the Simplex method. While it has very good convergence properties, but a rather slow convergence rate.

The basic idea is as follows:

0. A set of parameter points must be provided as an initial value. If all else fails, random points can be used.
1. Order the points by their respetive size of f
2. Compute the midpoint $\vec{a}_{\text{mid}}$ of all points except the worst, and reflect the worst point $\vec{a}_{N+1}$ at the midpoint yielding $\vec{a}_{\text{ref}}$.
   - a If $f(\vec{a}_{\text{ref}})$ fulfills some minimization criteria, replace one of the existing simplex points with $\vec{a}_{\text{ref}}$. Continue at step 1.
   - b Otherwise compute $\vec{a}_{\text{contr}}$ as a linear combination of the worst point of the simplex $\vec{a}_N$, and $\vec{a}_{\text{ref}}$. If $\vec{a}_{\text{contr}}$ is better than the worst point, replace the worst point. Continue at step 1.
   - c Otherwise compress the the simplex by moving the points $\vec{x}_1$ to $\vec{x}_N$ closer to $\vec{x}_0$ on their respective connecting lines. Continue at step 1.

Simplex Method (aka Nelder-Mead Method)

We will now discuss the work-horse of minimization algorithms, the Simplex method. While it has very good convergence properties, but a rather slow convergence rate.
The basic idea is as follows:

0 A set of parameter points must be provided as an initial value. If all else fails, random points can be used.
1 Order the points by their respetive size of f
2 Compute the midpoint $\vec{a}_{\text{mid}}$ of all points except the worst, and reflect the worst point $\vec{a}_{N+1}$ at the midpoint yielding $\vec{a}_{\text{ref}}$.
   a If $f(\vec{a}_{\text{ref}})$ fulfills some minimization criteria, replace one of the existing simplex points with $\vec{a}_{\text{ref}}$. Continue at step 1.
   b Otherwise compute $\vec{a}_{\text{contr}}$ as a linear combination of the worst point of the simplex $\vec{a}_N$, and $\vec{a}_{\text{ref}}$. If $\vec{a}_{\text{contr}}$ is better than the worst point, replace the worst point. Continue at step 1.
   c Otherwise compress the the simplex by moving the points $\vec{x}_1$ to $\vec{x}_N$ closer to $\vec{x}_0$ on their respective connecting lines. Continue at step 1.
3 If at any point the volume of the simplex falls below a given treshold, then stop.

Simplex Method (continued)

More details on the previous steps:

2 Compute the midpoint as

$$\vec{a}_{\mathrm{mid}} = \frac{1}{N}\sum_{n=0}^{N-1}\vec{a}_n$$

The reflection is computed as [default: $\alpha = 1$]

$$\vec{a}_{\mathrm{ref}} = (1+\alpha)\vec{a}_{\mathrm{mid}} - \alpha\vec{a}_N$$

Simplex Method (continued)

More details on the previous steps:

2 Compute the midpoint as

$$\vec{a}_{\text{mid}} = \frac{1}{N} \sum_{n=0}^{N-1} \vec{a}_n$$

The reflection is computed as [default: $\alpha = 1$]

$$\vec{a}_{\text{ref}} = (1+\alpha)\vec{a}_{\text{mid}} - \alpha \vec{a}_N$$

2a.1 If $\vec{a}_{\text{ref}}$ is better than $\vec{a}_0$, then replace $\vec{a}_N$ with $\vec{a}_{\text{ref}}$. Continue with step 1.

2a.1 alternative: Compute [default: $\epsilon = 2$]

$$\vec{a}_{\text{exp}} = (1+\epsilon)\vec{a}_{\text{mid}} - \epsilon \vec{a}_N$$

and use in step 2a the better of the two ($\vec{a}_{\text{ref}}$ or $\vec{a}_{\text{exp}}$). Continue with step 1.

Simplex Method (continued)

More details on the previous steps:

2a.2 If $\vec{a}_{\text{ref}}$ is better than the second worst point $\vec{a}_{N-1}$, then replace the worst point by $\vec{a}_{\text{ref}}$.

Simplex Method (continued)

More details on the previous steps:

2a.2 If $\vec{a}_{\mathrm{ref}}$ is better than the second worst point $\vec{a}_{N-1}$, then replace the worst point by $\vec{a}_{\mathrm{ref}}$.

2b Let $\vec{a}_{\mathrm{tmp}}$ be the better of $\vec{a}_{\mathrm{ref}}$ and $\vec{a}_{\mathrm{N}}$. Compute [default: $\gamma = 1/2$]

$$\vec{a}_{\mathrm{contr}} = \gamma \vec{a}_{\mathrm{mid}} + (1-\gamma)\vec{a}_{\mathrm{tmp}}\,.$$

If $\vec{a}_{\mathrm{contr}}$ is better than the worst point, replace the latter. Continue at step 1.

Simplex Method (continued)

More details on the previous steps:

2a.2 If $\vec{a}_{\text{ref}}$ is better than the second worst point $\vec{a}_{N-1}$, then replace the worst point by $\vec{a}_{\text{ref}}$.

2b Let $\vec{a}_{\text{tmp}}$ be the better of $\vec{a}_{\text{ref}}$ and $\vec{a}_{\text{N}}$. Compute [default: $\gamma = 1/2$]

$$\vec{a}_{\text{contr}} = \gamma \vec{a}_{\text{mid}} + (1-\gamma)\vec{a}_{\text{tmp}} \, .$$

If $\vec{a}_{\text{contr}}$ is better than the worst point, replace the latter. Continue at step 1.

2c Compress the entire simplex [default: $\kappa = 1/2$]

$$\vec{a}_n = \kappa \vec{a}_0 + (1-\kappa)\vec{a}_n \qquad \forall n = 1, \ldots, N$$

Simplex Method (continued)

More details on the previous steps:

2a.2 If $\vec{a}_{\mathrm{ref}}$ is better than the second worst point $\vec{a}_{N-1}$, then replace the worst point by $\vec{a}_{\mathrm{ref}}$.

2b Let $\vec{a}_{\mathrm{tmp}}$ be the better of $\vec{a}_{\mathrm{ref}}$ and $\vec{a}_{\mathrm{N}}$. Compute [default: $\gamma = 1/2$]

$$\vec{a}_{\mathrm{contr}} = \gamma \vec{a}_{\mathrm{mid}} + (1-\gamma)\vec{a}_{\mathrm{tmp}}\,.$$

If $\vec{a}_{\mathrm{contr}}$ is better than the worst point, replace the latter. Continue at step 1.

2c Compress the entire simplex [default: $\kappa = 1/2$]

$$\vec{a}_n = \kappa \vec{a}_0 + (1-\kappa)\vec{a}_n \qquad \forall n = 1, \ldots, N$$

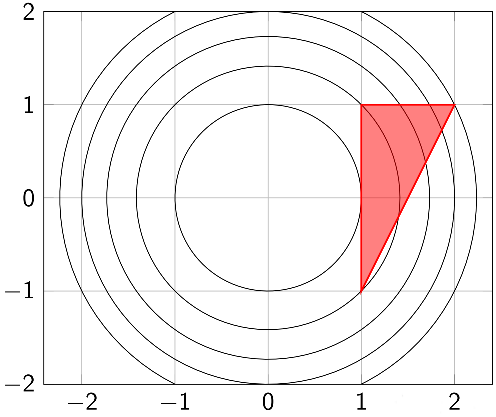
0 The $N+1$ initial points must span a simplex. Therefore, care must be taken that they do not lie in a linear subspace of the simplex. This is similar to picking points in a plane when constructing a 3D volume.

Simplex Method: Properties

- The method is very robust against problems such as overshooting.
- It will usually converge toward a close-by minimum.
- However, the convergence rate is smaller than for many other methods, which e.g. work on more specialised problems.
- No guarantee is given that the simplex method converges toward the global minimum. Possible relief comes in the form of several starting simplex obtained using Monte-Carlo methods.

Example

$$f(x, y) = x^2 + y^2$$



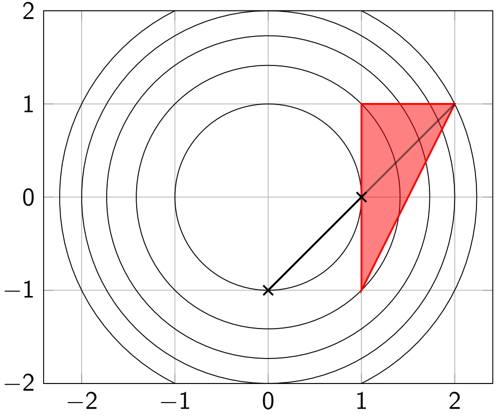
$a_0 = (+1, -1) \quad f(a_0) = 2$

$a_1 = (+1, +1) \quad f(a_1) = 2$

$a_2 = (+2, +1) \quad f(a_2) = 5$

Example

$$f(x, y) = x^2 + y^2$$



$a_0 = (+1, -1) \quad f(a_0) = 2$

$a_1 = (+1, +1) \quad f(a_1) = 2$

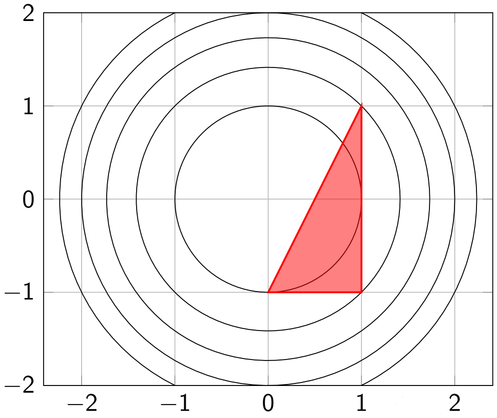
$a_2 = (+2, +1) \quad f(a_2) = 5$

$a_{\text{mid}} = (+1, 0)$

$a_{\text{ref}} = (0, -1) \quad f(a_{\text{ref}}) = 1$

Example

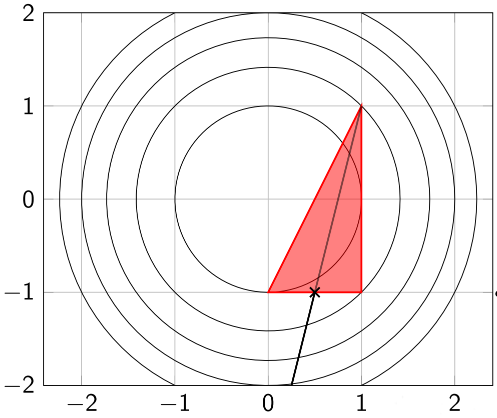
$$f(x, y) = x^2 + y^2$$



$$\begin{aligned} a_0 &= (+0, -1) & f(a_0) &= 1 \\ a_1 &= (+1, -1) & f(a_1) &= 2 \\ a_2 &= (+1, +1) & f(a_2) &= 2 \end{aligned}$$

Example

$$f(x, y) = x^2 + y^2$$



$a_0 = (+0, -1) \quad f(a_0) = 1$

$a_1 = (+1, -1) \quad f(a_1) = 2$

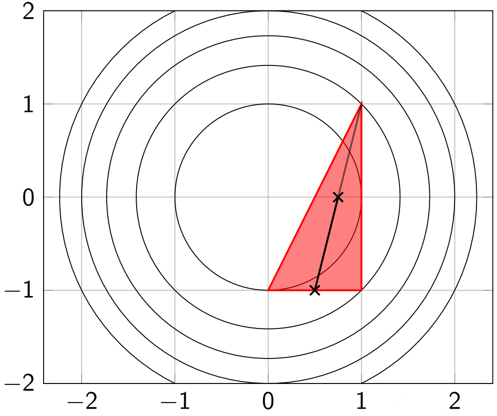
$a_2 = (+1, +1) \quad f(a_2) = 2$

$a_{\text{mid}} = (+1/2, -1)$

$a_{\text{ref}} = (0, -3) \quad f(a_{\text{ref}}) = 9$

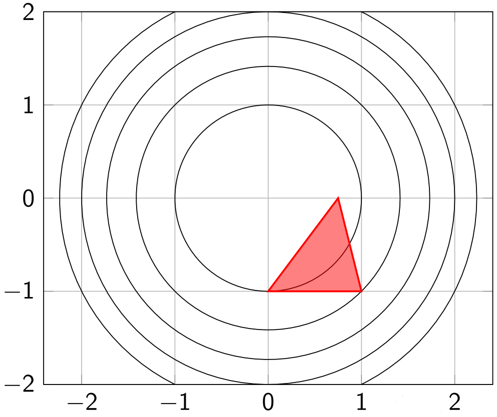
Example

$$f(x, y) = x^2 + y^2$$



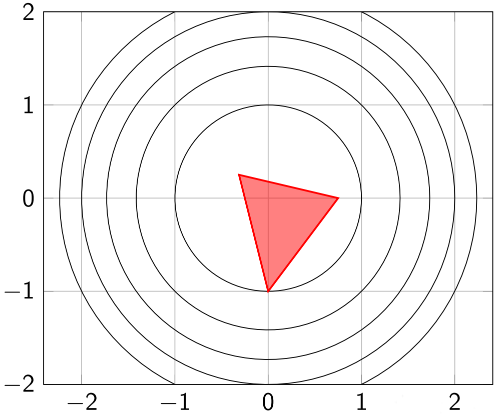
Example

$$f(x, y) = x^2 + y^2$$



Example

$$f(x, y) = x^2 + y^2$$



Least-squares Problems

The target function is called the residue $r(a_1, \ldots, a_N)$ with N parameters.

$$r_k \equiv y(\vec{a}, \vec{x}_k) - y_k$$

Here k denotes one of the K possible coordinate on a curve, with (external, e.g. experimental) inputs $(\vec{x}_k, y_k)$.

The problem now aims to minimize

$$f(\vec{a}) \equiv \sum_k^K |r_k|^2$$

The least-squares problem arises from the case of a Gaussian likelihood function if all uncertainties are equal. It is a very good example for understanding a non-linear problem through linearization.

Gauss-Newton Method

This iterative method requires partial derivatives of the resiudes r_k with respect to the parameters a_n:

$$r'_{k,n} \equiv \frac{\partial r(\vec{a}, \vec{x})}{\partial a_n}\bigg|_{\vec{x}=\vec{x}_k}$$

The algorithm now involves the following quantities:

$$J = \begin{pmatrix} r'_{1,1} & r'_{1,2} & \cdots & r'_{1,n} \\ r'_{2,1} & r'_{2,2} & \cdots & r'_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ r'_{k,1} & r'_{k,2} & \cdots & r'_{k,n} \end{pmatrix} \qquad \vec{r} = (r_1, \ldots, r_k)^T$$

Here J is the Jacobi matrix of the residue.

Gauss-Newton Method (cont'd)

0 As always, we will be requiring a starting point $\vec{a}_0$ in parameter space.

1 Update the current point:

$$\vec{a}_{i+1} = \vec{a}_i - (J^T \cdot J)^{-1} \cdot J \cdot \vec{r}$$

2 If $||\vec{a}_{i+1} - \vec{a}_i|| < T$, where T is some a-prior threshold, we stop. Otherwise, continue with step 1.

Some comments

- The literature usually recommends to compute the auxiliary variable $\vec{s}$ via:

  $$(J^T \cdot J) \cdot \vec{s} = J^T \cdot \vec{r}$$

  The above linear system of equations can be solved with known methods.
- Since $(J^T \cdot J)$ is symmetric, it is a good idea to use Cholesky decomposition.

Derivation

Why does this work? Necessary and sufficient conditions for an optimum are:

$$\vec{g} \equiv \frac{\partial f(\vec{a})}{\partial a_n} \overset{!}{=} 0 \qquad \text{and} \qquad \det h \overset{!}{\neq} 0 \qquad \text{with} \quad h \equiv \frac{\partial^2 f}{\partial a_n \, \partial a_{n'}}$$

One could therefore use Newton's method to find the zeros of the gradient:

$$\vec{a}_{i+1} = \vec{a}_i - (h)^{-1} \vec{g}$$

Express $\vec{g}$ and h in terms of the $r'_{k,n}$:

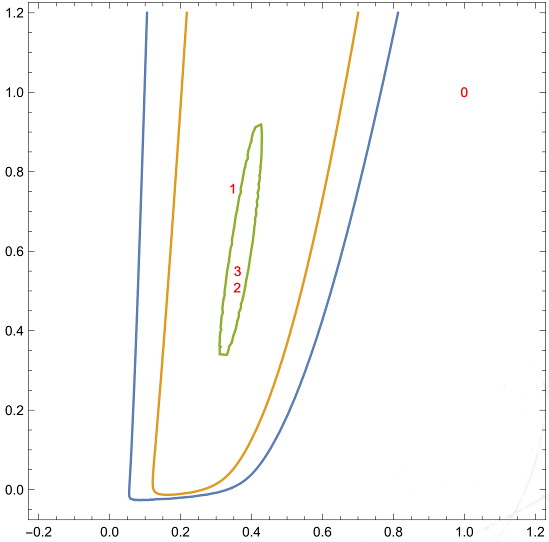
$$g_n = \sum_k^K 2 \frac{\partial r_k}{\partial a_n} r_k \qquad H_{n,n'} = \sum_k^K 2 \frac{\partial^2 r_k}{\partial a_n \, \partial a_{n'}} r_k + 2 \frac{\partial r_k}{\partial a_n} \frac{\partial r_k}{\partial a_{n'}}$$

$$= 2J \cdot \vec{r} \qquad\qquad = \left[\sum_k^K 2 \frac{\partial^2 r_k}{\partial a_n \, \partial a_{n'}} \right] + 2J^T \cdot J$$

Assuming the second derivatives are small compared to the square first-deriv. term, than we can neglect hem.

Example

$$y(x, a_1, a_2) = \frac{a_1 x}{a_2 + x} \qquad \text{7 data points}$$



Levenberg-Marquardt Method

The Levenberg-Marquardt method arises from a modification to the Gauss-Newton method.

The adjustment length is attenuated through a dampening parameter λ. Steers adjustment away from Gauss-Newton direction to the gradient's direction.

0 As always, we will be requiring a starting point $\vec{a}_0$ in parameter space.

1 Update the current point:

$$\vec{a}_{i+1} = \vec{a}_i - (J^T \cdot J + \lambda I)^{-1} \cdot J \cdot \vec{r}$$

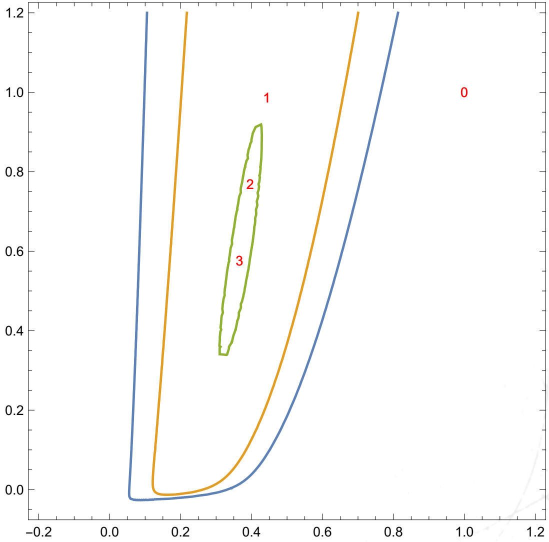
where I is the unit matrix in $N \times N$ and λ a real-valued parameter.

2 If $||\vec{a}_{i+1} - \vec{a}_i|| < T$, where T is some a-prior threshold, we stop. Otherwise, continue with step 1.

The optimal choice of λ is specific to the problem.

Example

$$y(x, a_1, a_2) = \frac{a_1 x}{a_2 + x} \qquad \text{7 data points}$$



Backup