Trust Region Methods

Numerical Optimization

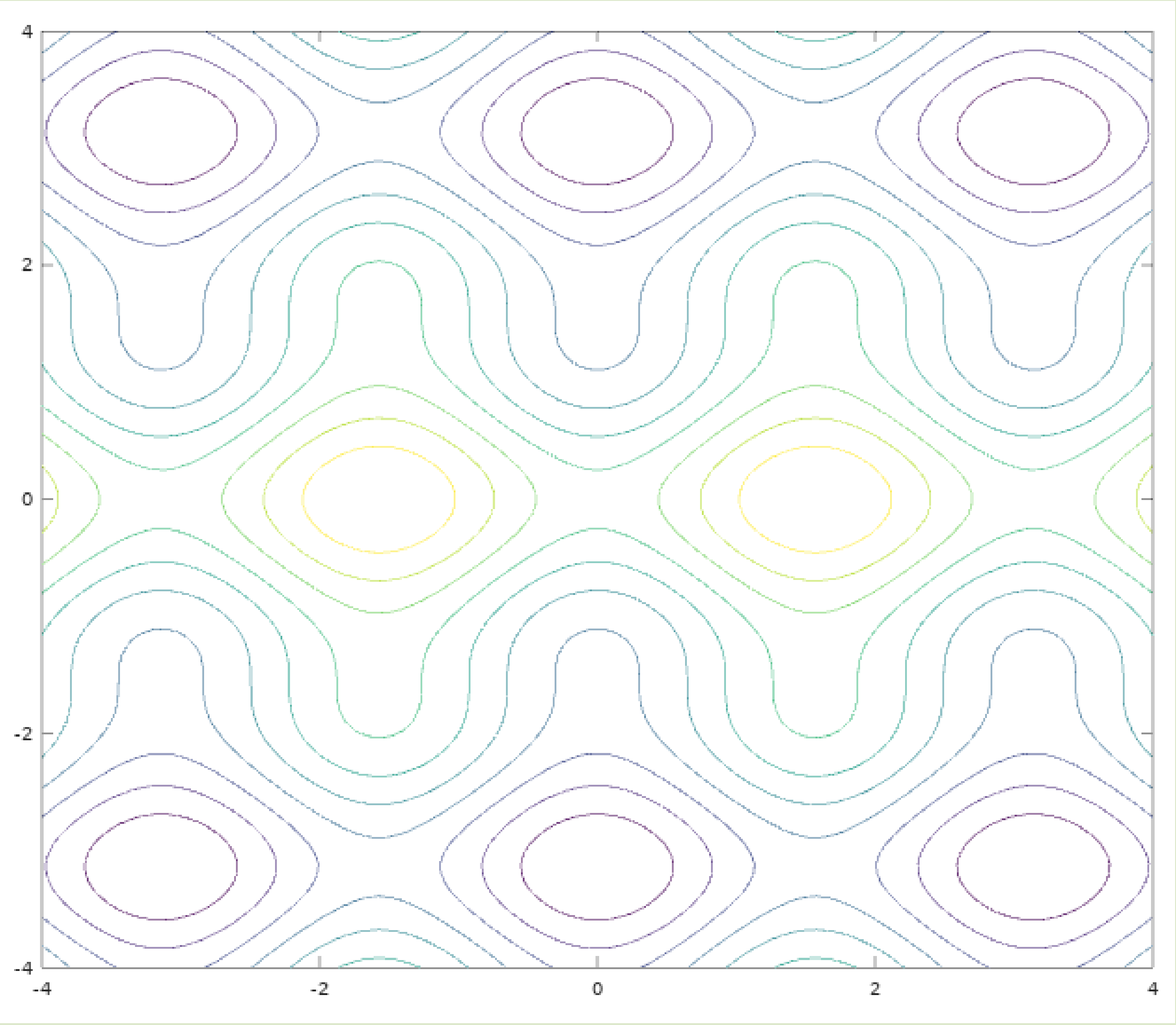
Prepared by LimCK

Line Search vs Trust Region

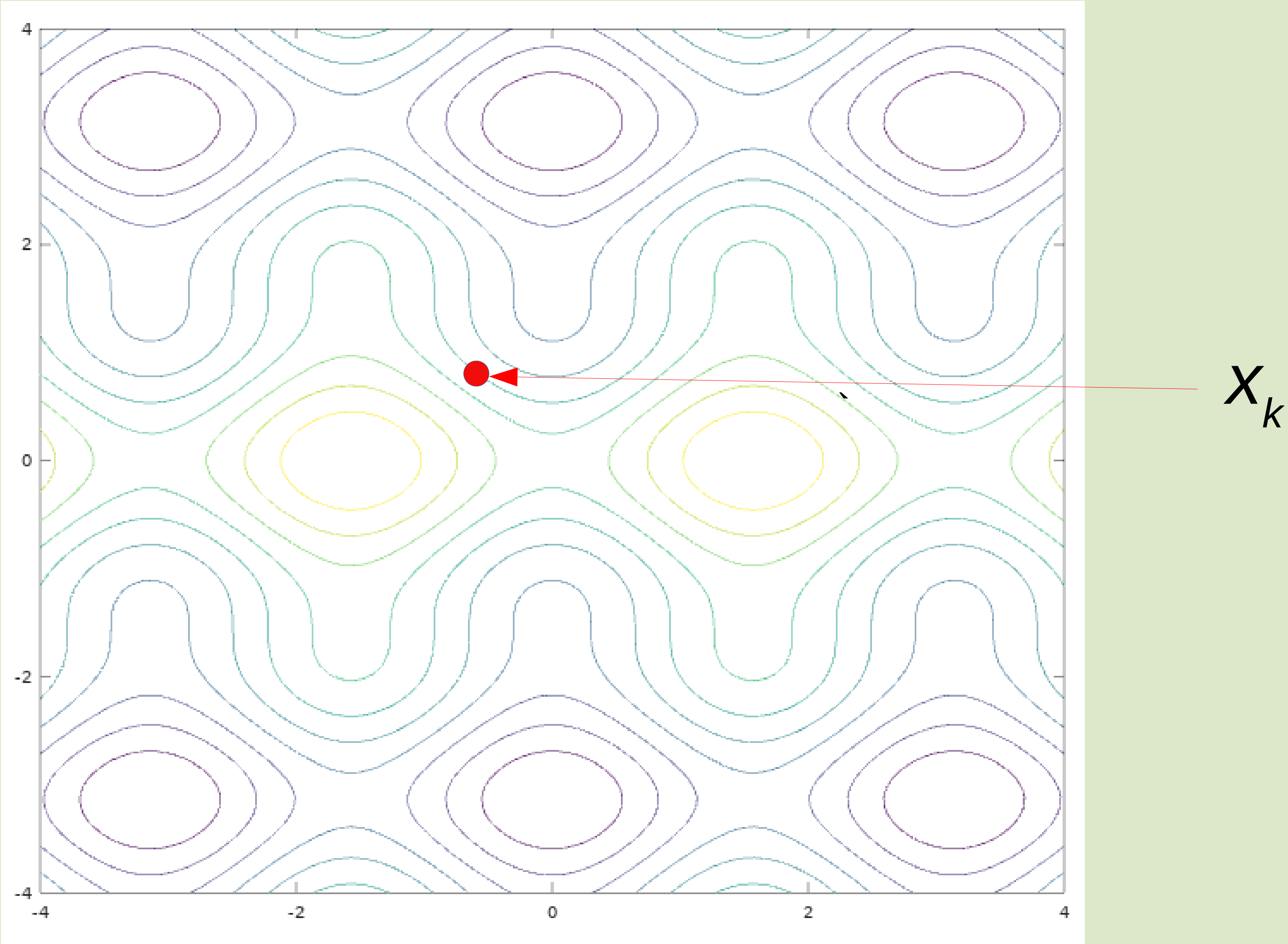
- Line search – first determine direction *p*, then find a suitable step length *α*.
- Trust Region:

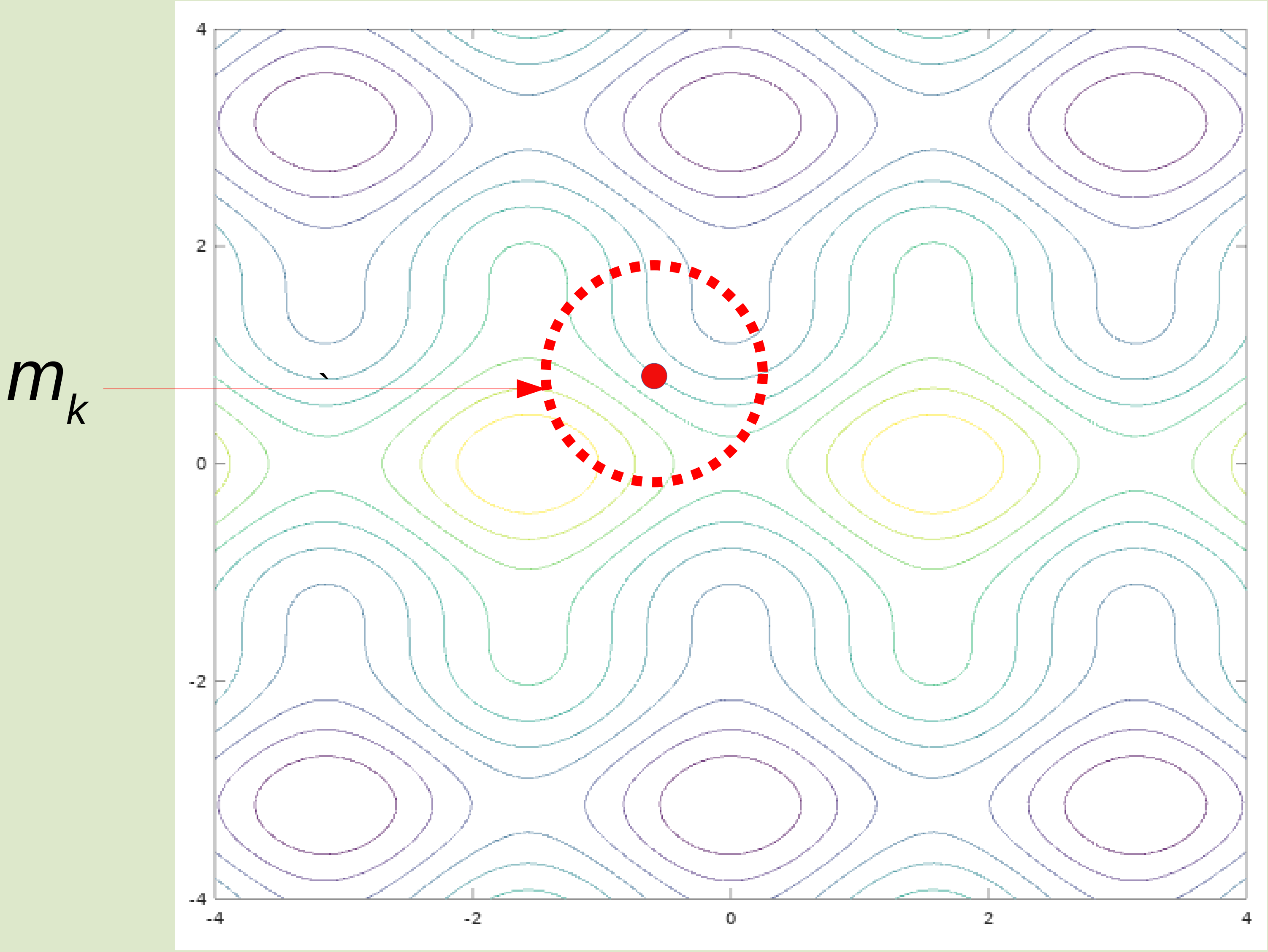
  1. At current search point (x_k), define a region which a certain model (m_k) can approximate the original objective function *f*, to some extent
  2. Choose the step to the minimizer in this region.

  We assume that a quadratic model is adequate to model the original functions.

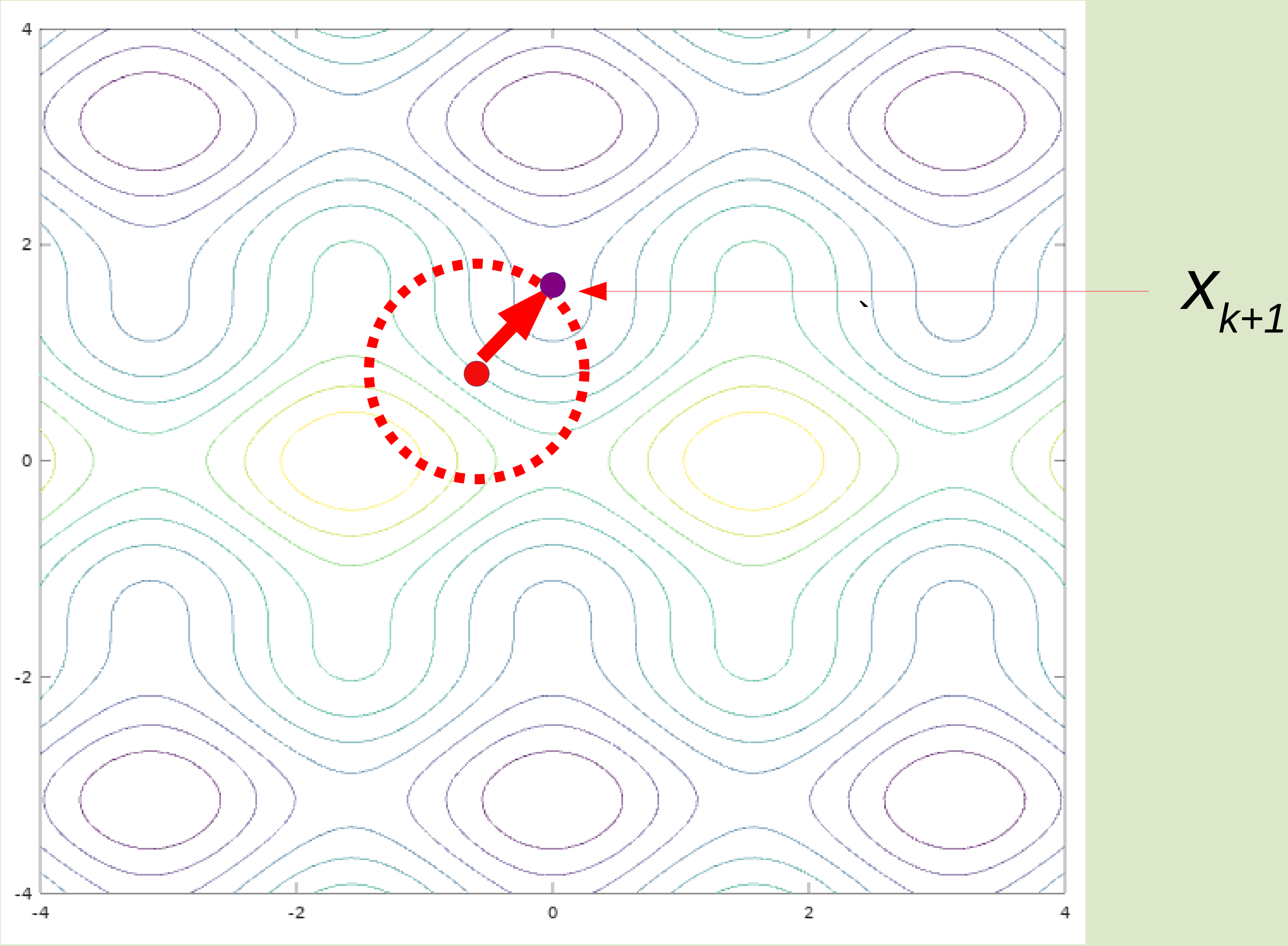


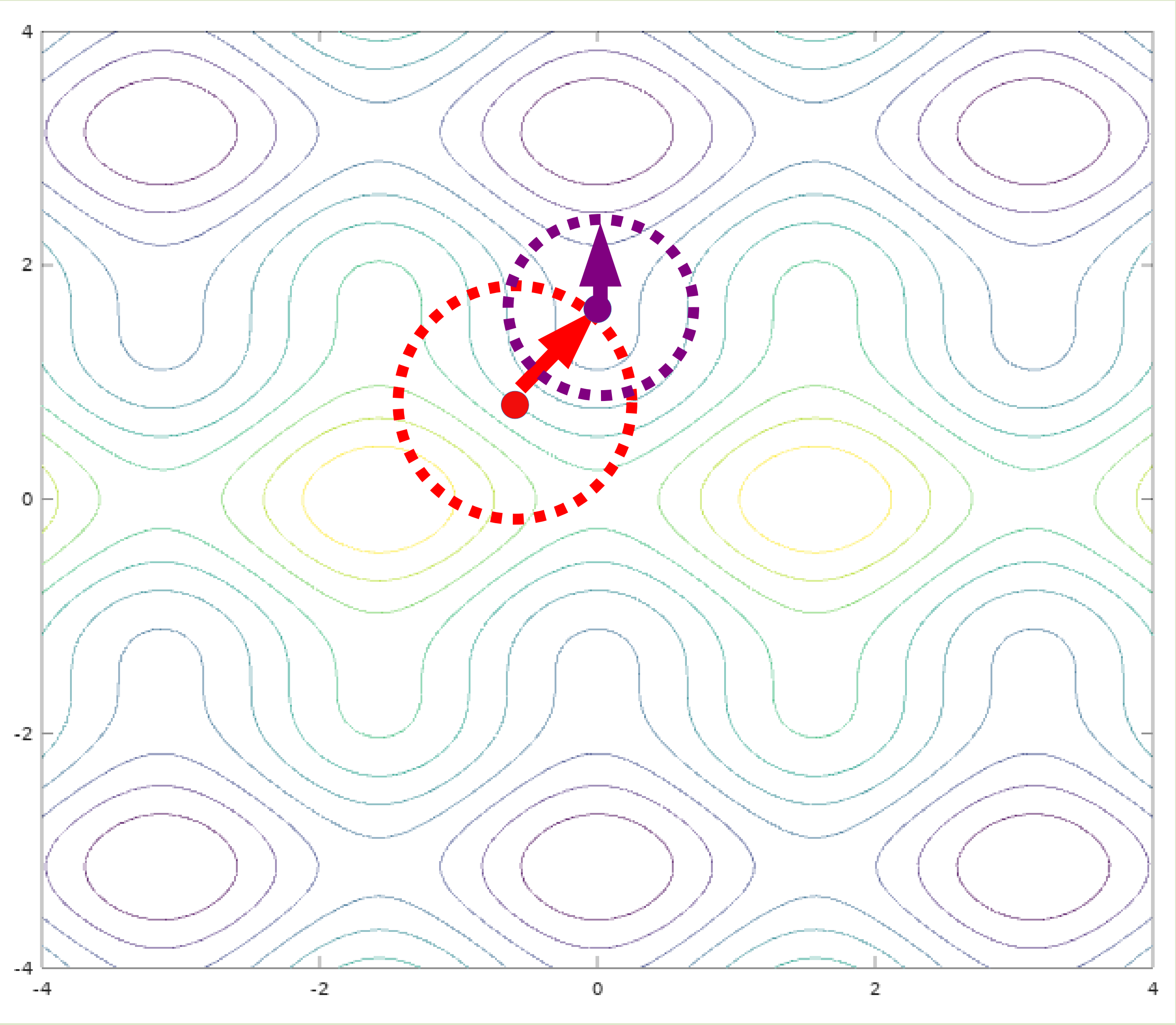
4
2
0
-2
-4
-4
-2
0
2
4


4
2
0
-2
-4
-4
-2
0
2
4
x_k



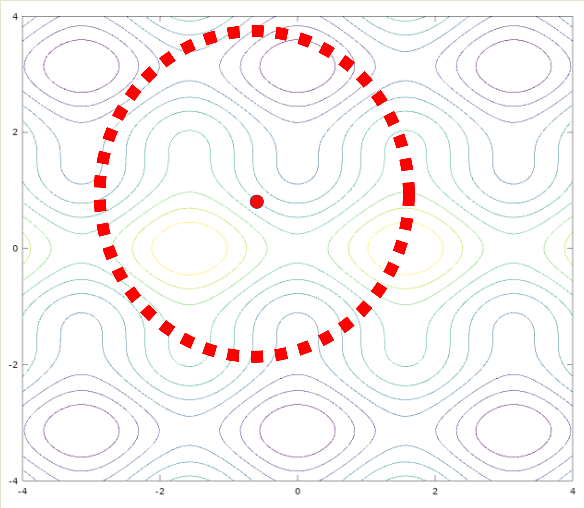
m_k
4
2
0
-2
-4
-4
-2
0
2
4


4
2
0
-2
-4
-4
-2
0
2
4
x_{k+1}

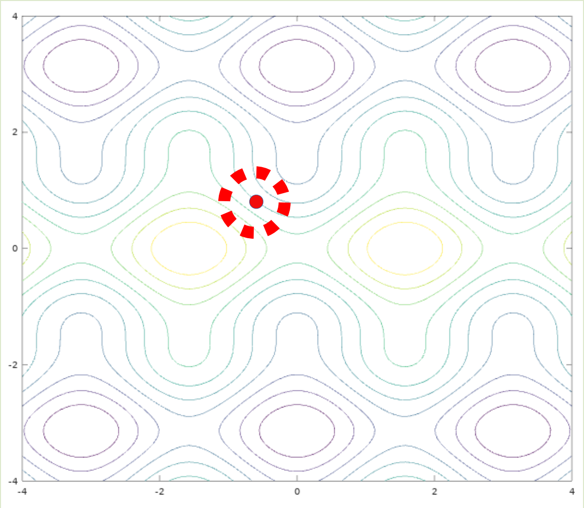

4
2
0
-2
-4
-4
-2
0
2
4

Trust Region – Basic Idea

- The size of the trust region (Δ_k) is critical:
  - If too large : the predicted (modelled) minimizer may be too far from expected (real) minimizer
  - If too small : small steps, slow.



Δ_k too large



Δ_k too small

Trust Region – Basic Idea

- May refer to performance of last step/iteration to determine the size of the region:
  - Good : increase region size
  - Fail : inadequate to model the region – reduce size

Trust Region – Basic Idea

With Taylor's expansion, assuming m_k as a quadratic model (works in many cases):

$$m_k(p) = f_k + g_k^T p + \frac{1}{2} p^T B_k p \tag{1}$$

$f(x_k)$

$g_k = \nabla f(x_k)$

B_k – Hessian or approximation

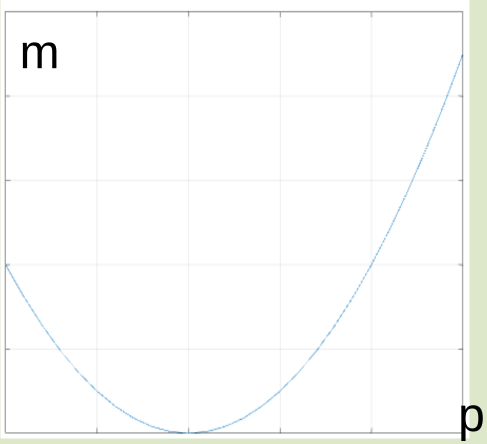
This model is especially accurate when ||p|| is small

Trust Region – Basic Idea

With Taylor's expansion, assuming m_k as a quadratic model (works in many cases):

$$m(p) = a + b\,p + c\,p^2$$

$$m_k(p) = f_k + g_k^T p + \tfrac{1}{2} p^T B_k p \qquad (1)$$



$f(x_k)$

$g_k = \nabla f(x_k)$

B_k – Hessian or approximation

This model is especially accurate when $||p||$ is small

Trust Region – Basic Idea

- To obtain each step, we seek a solution (p_k*) of the subproblem:

$$\min_{p \in \mathbb{R}^n} m_k(p) = f_k + g_k^T p + \tfrac{1}{2} p^T B_k p \qquad \text{s.t. } \|p\| \le \Delta_k \tag{2}$$

where $\Delta_k > 0$ is the trust-region radius

- If B_k is positive definite, the solution is easy to find:

$$g_k^T + 2 \cdot \frac{1}{2} p_k B_k = 0$$

$$p_k = -B_k^{-1} g_k^T \tag{3}$$

- However, if B_k is not positive definite, more computation is required.

Trust Region Algorithm

- Based on the computation of ρ:

$$\rho_k = \frac{f(x_k) - f(x_k + p_k)}{m_k(0) - m_k(p_k)} \tag{4}$$

actual reduction

predicted reduction

Ratio between actual reduction and predicted reduction determine whether a model m_k is a good representation or not.

- ρ should not be negative.
- If negative or close to zero, shrink the Δ_k
- Close to 1 : *m* is a good model, may expend the region in next iteration.

Trust Region Algorithm

Initialization: k = 0 and $\tilde{\Delta}$ = upper bound of the radius of the trust region

while not converge {

obtain p_k by solving trust region sub-problem $m_k(p_k) = f_k + g_k^T p_k + \frac{1}{2} p_k^T B_k p_k$

evaluate ρ_k, the ratio of actual reduction over predicted reduction

if ρ_k is too small

consider a smaller radius Δ

else if ρ_k is large enough and taking full step is allowed

consider to increase the radius Δ

else

consider current radius Δ

if ρ_k is larger then a threshold

accept this model and take this move

else

try again with a new model (smaller radius)

increase k by 1

}

Trust Region Algorithm

Initialization: k = 0 and $\tilde{\Delta}$ = upper bound of the radius of the trust region

for k = 0, 1, 2, 3, ...

p_k = solution_of_trust_region_sub-problem( )

$\rho_k = (f(x_k) - f(x_k + p_k)) / (m_k(0) - m_k(p_k))$

if $\rho_k < \eta_1$

$\Delta_{k+1} = t_1 \Delta_k$

else if $\rho_k > \eta_2$ and $\| p_k \| = \Delta_k$

$\Delta_{k+1} = \min (t_2 \Delta_k , \tilde{\Delta})$

else

$\Delta_{k+1} = \Delta_k$

if $\rho_k > \eta_3$

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

else

$x_{k+1} = x_k$

end (for)

Typical values

$\eta_1 = 0.25$

$\eta_2 = 0.75$

$0 \le \eta_3 \le \eta_1$

$t_1 = 0.25$

$t_2 = 2$

Cauchy Point

- Cauchy point – strategy to solve the trust region subproblem.
- Like line search method, optimal solution *p** is not required, but we just look for approximate solution p_k that lies within the trust region and gives a sufficient reduction
- The sufficient reduction can be quantified in terms of the Cauchy point, which we denote by:

$$p_k^c$$

Cauchy Point Calculation

- Consider the linear model of eq (2):

$$l(p) = f_k + \nabla f_k^T p = f_k + g_k^T p \tag{5}$$

- The gradient of this linear model is g_k. A set of points along this direction:

$$p_k^{\mathrm{S}} = -\frac{\Delta_k}{\|g_k\|} g_k \tag{6}$$

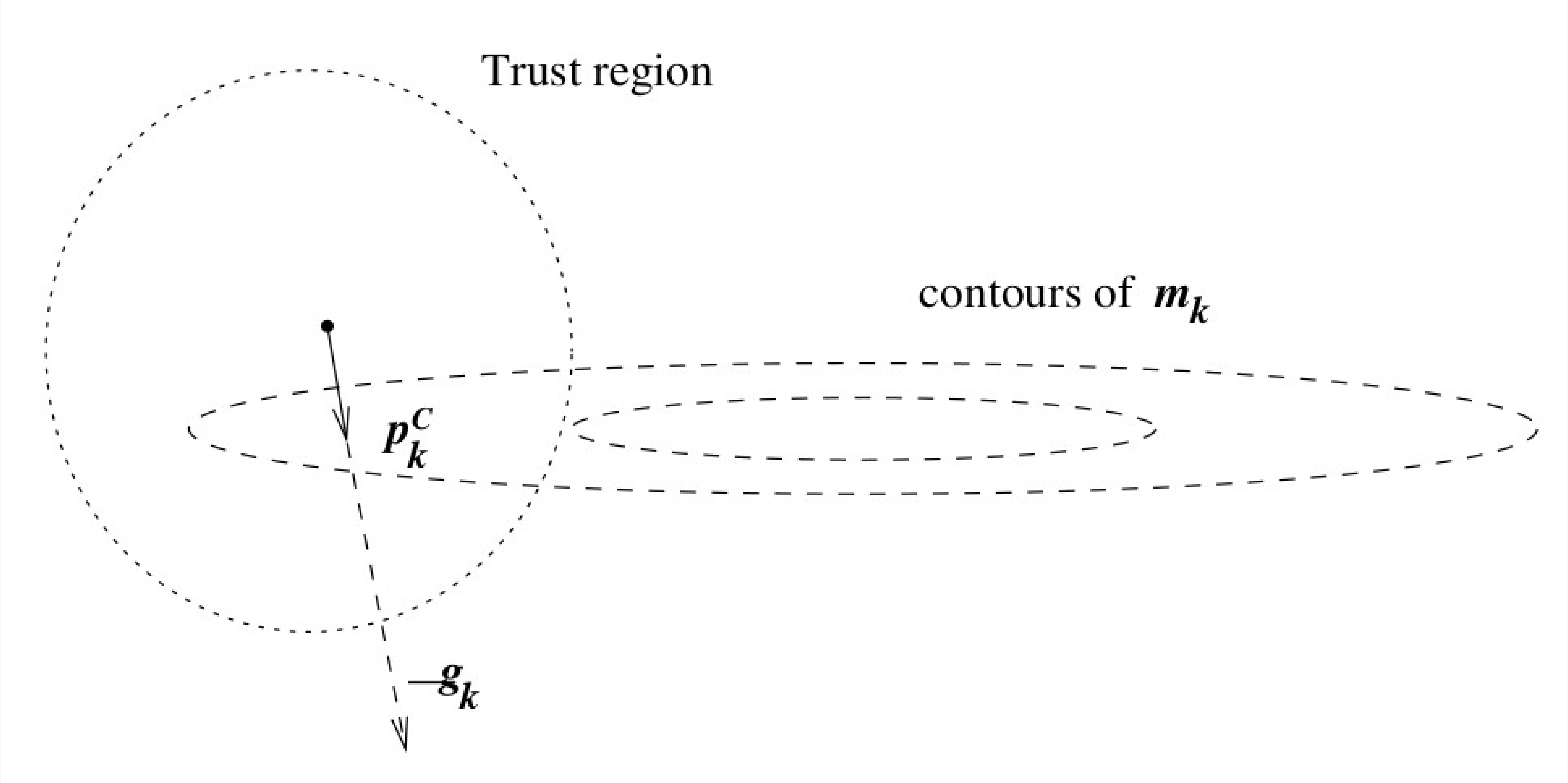
- Cauchy point is a specific point along this direction given by Cauchy step:

$$p_k^{\mathrm{C}} = -\tau_k \frac{\Delta_k}{\|g_k\|} g_k \tag{7}$$

$$\tau_k = \begin{cases} 1 & \text{if } g_k^T B_k g_k \leq 0; \\ \min\left(\|g_k\|^3/(\Delta_k g_k^T B_k g_k), 1\right) & \text{otherwise.} \end{cases}$$

- Compute this Cauchy step is inexpensive – limited matrix ops

Cauchy Point Calculation



Improving on the Cauchy Point

- Cauchy Point provides sufficient reduction with low cost
- However, performance can be poor in some cases.
- Improvement strategy: include the information provided by B_k.
- Example:
  - Dogleg method
  - Conjugated Gradient Steihaug's Method

Dogleg Method

- For the case if B is positive definite.
- Recall eq (3):

$$p^B = -B^{-1} g^T$$

- However, this only work if Δ is big relative to p^B, i.e. $\Delta \geq || p^B ||$
- How about if Δ is small or intermidiate?

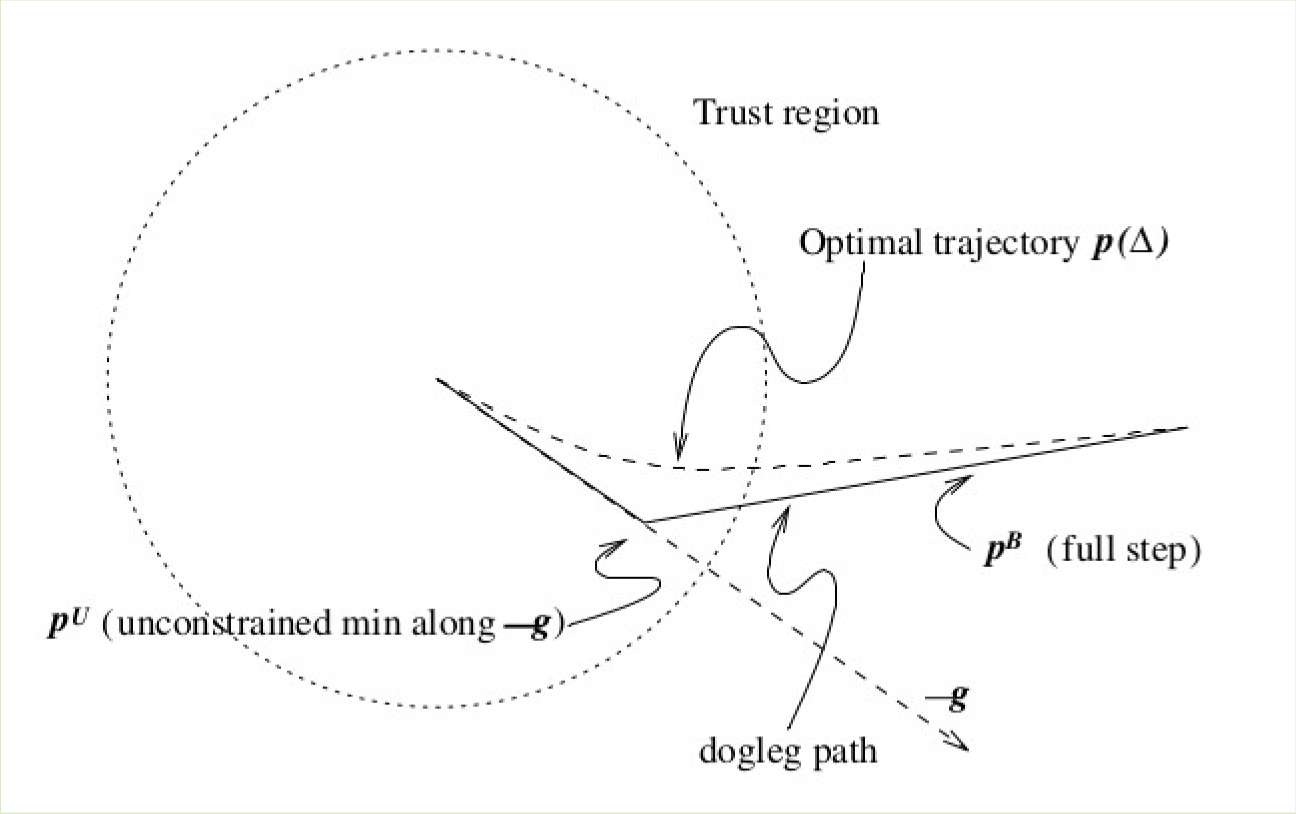
Dogleg Method

- If Δ is small: not much different between a curve and a straight line – the quadratic term in eq (2) can be dropped.
- Therefore, from the Cauchy Point calculation, i.e. eq (5), we obtained eq (6):

$$p^*(\Delta) \approx -\Delta \frac{g}{\|g\|}$$

Dogleg Method

- If Δ is Intermediate: the solution is a curve p(Δ).
- Instead of finding this curve, we find approximation: 2 line segments p^U and p^B



Dogleg Method

- The first line segment is p^U:

$$p^{\mathrm{U}} = -\frac{g^T g}{g^T B g} g \qquad (8)$$

- The second line segment is p^B.

Dogleg Method

- The first line segment is p^U:

$$p^{\mathrm{U}} = -\frac{g^T g}{g^T B g} g \tag{8}$$

- The second line segment is p^B.
- Formally, the trajectory is represented as:

$$\tilde{p}(\tau) = \begin{cases} \tau p^{\mathrm{U}}, & 0 \le \tau \le 1, \\ p^{\mathrm{U}} + (\tau - 1)(p^{\mathrm{B}} - p^{\mathrm{U}}), & 1 \le \tau \le 2. \end{cases} \tag{9}$$

for $\tau \in [0, 2]$

Dogleg Method

- The path intersects the trust region boundary at only one point if $p^B \geq \Delta$, i.e. when $||p|| = \Delta$
- If $\Delta \geq p^B$, always take p^B.
- Otherwise, the value of τ can be compute by solving:

$$\| p^U + (\tau - 1)(p^B - p^U) \|^2 = \Delta^2 \tag{10}$$

Any Question?

Exercise (Q1)

- For trust region algorithm, the typical value for η_1 is 0.25 and $0 \leq \eta_3 \leq \eta_1$. What is the effect if :
  - we set η_1 to a smaller value, say 0.1?
  - We set $\eta_3 > \eta_1$?

Exercise (Q2)

- A Branin Function is given as follow:

$$f(x) = 1.4(x_2 - 0.135x_1^2 + 1.3x_1 - 10)^2 + 3.642\cos(x_1) + 6$$

- Given that the initial value of x is (-2,2). With default values for trust region algorithm, compute 1 iteration of trust region algorithm.
- There are a few global minimizers for this function. Write an Octave program to find one with the same settings as above. Plot the contour in the range [-5,15] and [0,15] for x_1 and x_2 respectively.
- Which minimum you will reach if you start at:
  - (-2,9)
  - (15,0)