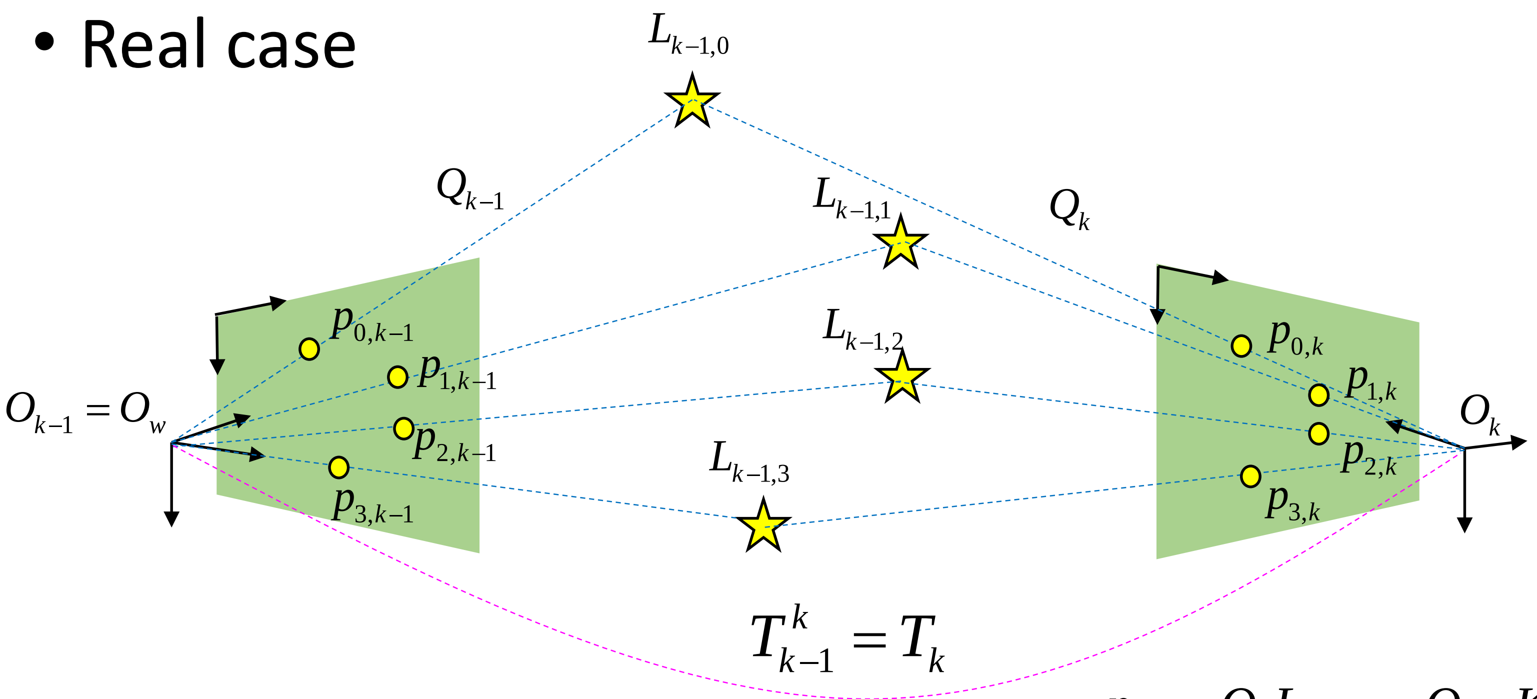
Numerical Optimization

2016.12.12

김태원

Feature based method

- Real case



$$p_{i,k} = Q_k L_{k-1,i}, \quad Q = K\left[R \mid t\right]$$

$$\hat{p}_{i,k} = K\left[R_{\xi_k} \mid t_{\xi_k}\right] L_{k-1,i}$$

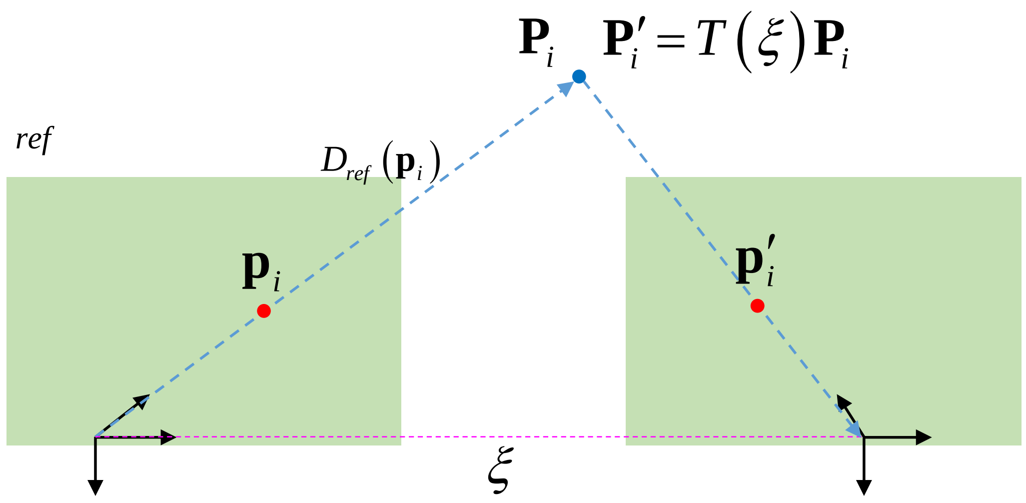
$$\xi_{k-1} = 0$$

$$\underset{\xi_k}{\operatorname{argmin}}\left(\sum\left(p_{i,k} - \hat{p}_{i,k}\right)^2\right)$$

$$T_{k-1}^{k} = \exp\left(\xi_k\right)$$

Direct method

$$E(\xi)=\sum_i\Big(I_{ref}(\mathbf{p}_i)-I\big(\omega\big(\mathbf{p}_i,D_{ref}(\mathbf{p}_i),\xi\big)\big)\Big)^2$$



$$E(\xi)=\sum_i\Big(I_{ref}(\mathbf{p}_i)-I\big(\omega(\mathbf{P}_i,\xi)\big)\Big)^2$$

$$E(\xi)=\sum_i\Big(I_{ref}(\mathbf{p}_i)-I\big(\omega(\mathbf{P}_i')\big)\Big)^2$$

$$E(\xi)=\sum_i\Big(I_{ref}(\mathbf{p}_i)-I(\mathbf{p}_i')\Big)^2$$

$\mathbf{p}_i\in\mathbf{R}^2$ is pixel coordinates

$I_{ref}(\mathbf{p}_i)\in\mathbf{R}$ is intensity

$D_{ref}(\mathbf{p}_i)\in\mathbf{R}$ is depth

Cost functions

- Feature based method

Find pair → Motion estimation

$$\underset{\xi}{\operatorname{argmin}}\left(\sum_{i}^{N}\left(L_{i,k}-T_{k-1}^{k}L_{i,k-1}\right)^{2}\right)$$

$$p_{i,k}=Q_k L_{k-1,i},\quad Q=K\left[R\,|\,t\right]$$

$$\hat{p}_{i,k}=K\left[R_{\xi_k}\,|\,t_{\xi_k}\right]L_{k-1,i}$$

$$E\left(\xi_k\right)=\sum\left(p_{i,k}-\hat{p}_{i,k}\right)^{2}$$

$$\xi_k^{*}=\underset{\xi_k}{\operatorname{argmin}}\left(E\left(\xi_k\right)\right)$$

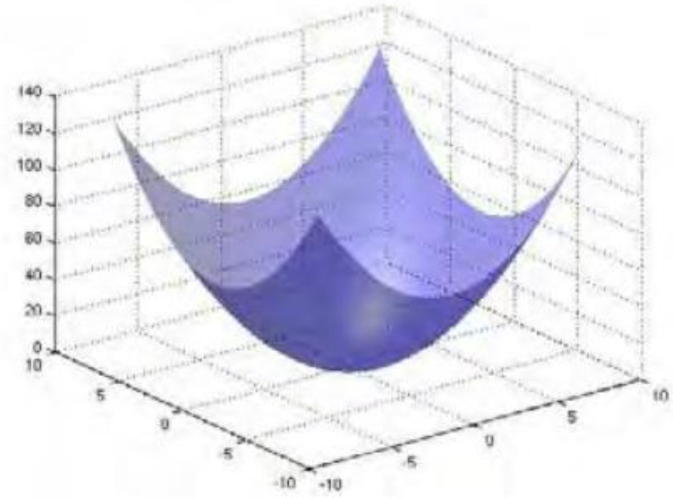
- Direct method

Motion estimation

$$E\left(\xi\right)=\sum_{i}\left(I_{ref}\left(\mathbf{p}_i\right)-I\left(\omega\left(\mathbf{p}_i,D_{ref}\left(\mathbf{p}_i\right),\xi\right)\right)\right)^{2}$$

$$\xi^{*}=\underset{\xi}{\operatorname{argmin}}\left(E\left(\xi\right)\right)$$

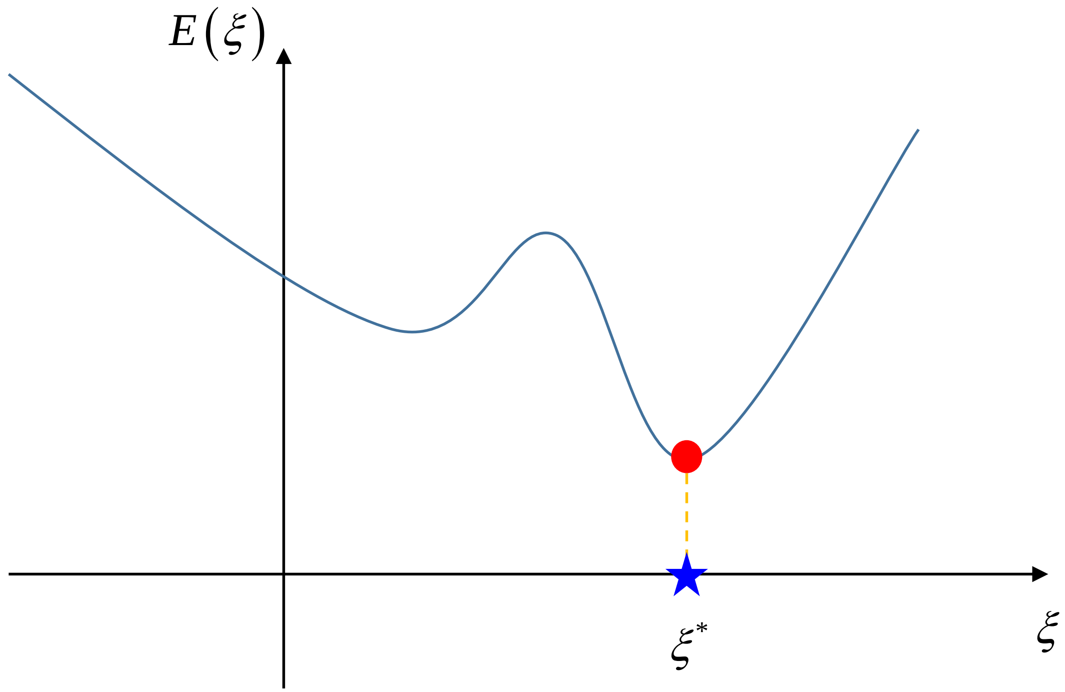
Desirable Characteristics of a Cost Function



- Scalar
- Clearly defined (preferably unique) maximum or minimum
  - Local
  - Global
- Preferably *positive-definite* (i.e., always a positive number)

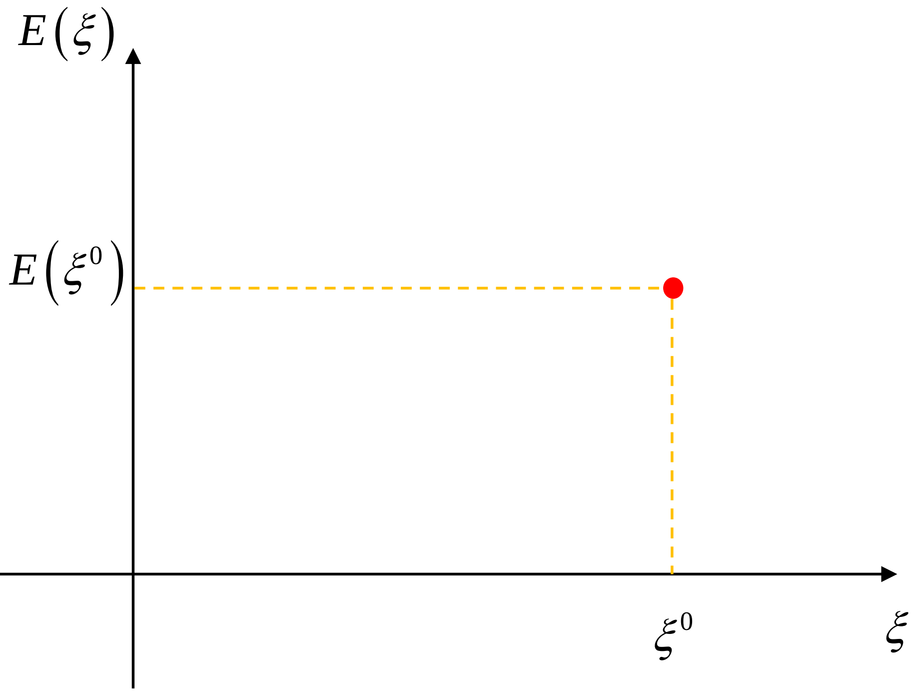
Goal of Numerical Optimization

$$\xi^* = \underset{\xi}{\operatorname{argmin}} \left(E(\xi) \right)$$



Search Problem

$$\xi^* = \underset{\xi}{\operatorname{argmin}}\left(E\left(\xi\right)\right)$$



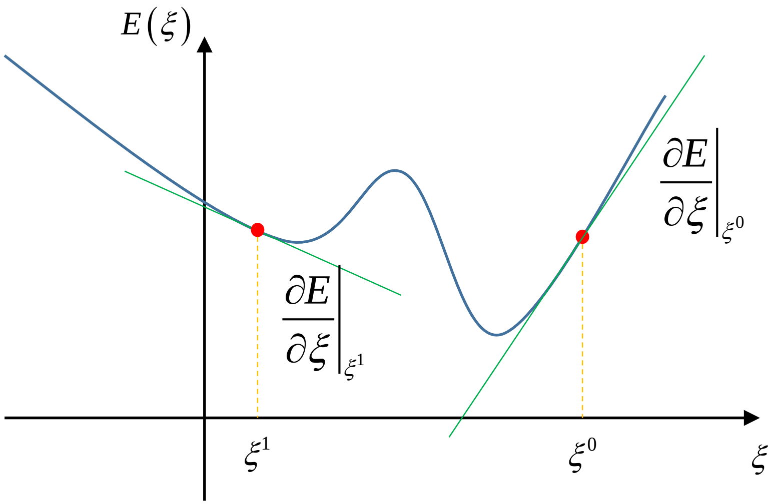
Two Approaches to Numerical Minimization

1. Gradient Search

2. Gradient-free search

Gradient

- Gradient



ex) $E(\xi) = \xi^3$

$$\frac{\partial E}{\partial \xi} = 3\xi^2$$

$$\left.\frac{\partial E}{\partial \xi}\right|_{\xi^0} = 3\left(\xi^0\right)^2$$

- Jacobian

$$\xi = \begin{bmatrix} \xi_1 & \xi_2 & \xi_3 & \xi_4 & \xi_5 & \xi_6 \end{bmatrix}^T$$

$$\frac{\partial E}{\partial \xi} = \begin{bmatrix} \frac{\partial E}{\partial \xi_1} & \frac{\partial E}{\partial \xi_2} & \frac{\partial E}{\partial \xi_3} & \frac{\partial E}{\partial \xi_4} & \frac{\partial E}{\partial \xi_5} & \frac{\partial E}{\partial \xi_6} \end{bmatrix}$$

ex) $E(\xi) = \xi_1{}^2 + \xi_2{}^2 + \xi_3{}^2 + \xi_4{}^2 + \xi_5{}^2 + \xi_6{}^2$

$$\frac{\partial E}{\partial \xi} = \begin{bmatrix} 2\xi_1 & 2\xi_2 & 2\xi_3 & 2\xi_4 & 2\xi_5 & 2\xi_6 \end{bmatrix}$$

$$\xi^0 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}^T$$

$$\left.\frac{\partial E}{\partial \xi}\right|_{\xi^0} = \begin{bmatrix} 2 & 2 & 2 & 2 & 2 & 2 \end{bmatrix}$$

Gradient descent = Steepest Descent

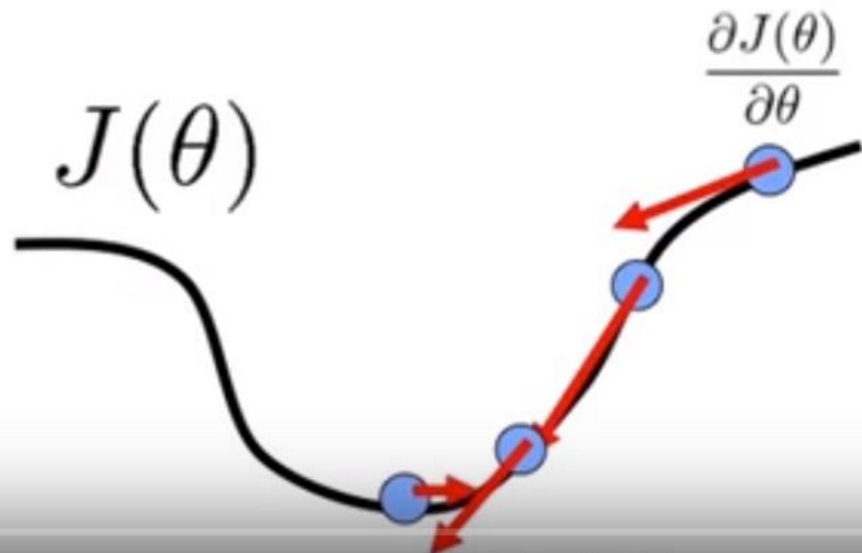
- Initialization
- Step size
  - Can change as a function of iteration
- Gradient direction
- Stopping condition

$$\texttt{Initialize}\ \theta$$

$$\texttt{Do}\ \{$$

$$\theta \leftarrow \theta - \alpha\, \nabla_\theta J(\theta)$$

$$\}\ \texttt{while}\ (\ \alpha \|\nabla J\| > \epsilon\ )$$



Example

$$\underline{\theta}^{k+1} \leftarrow \underline{\theta}^{k} - \alpha \nabla_{\underline{\theta}} J(\underline{\theta}^{k})$$

$$\alpha = 0.1$$

$$E(\xi) = (\xi - 5)^2 = \xi^2 - 10\xi + 25$$

$$\frac{\partial E}{\partial \xi} = 2\xi - 10$$

$$\xi^0 = 0 \qquad \xi^1 \leftarrow 0 - 0.1(-10)$$

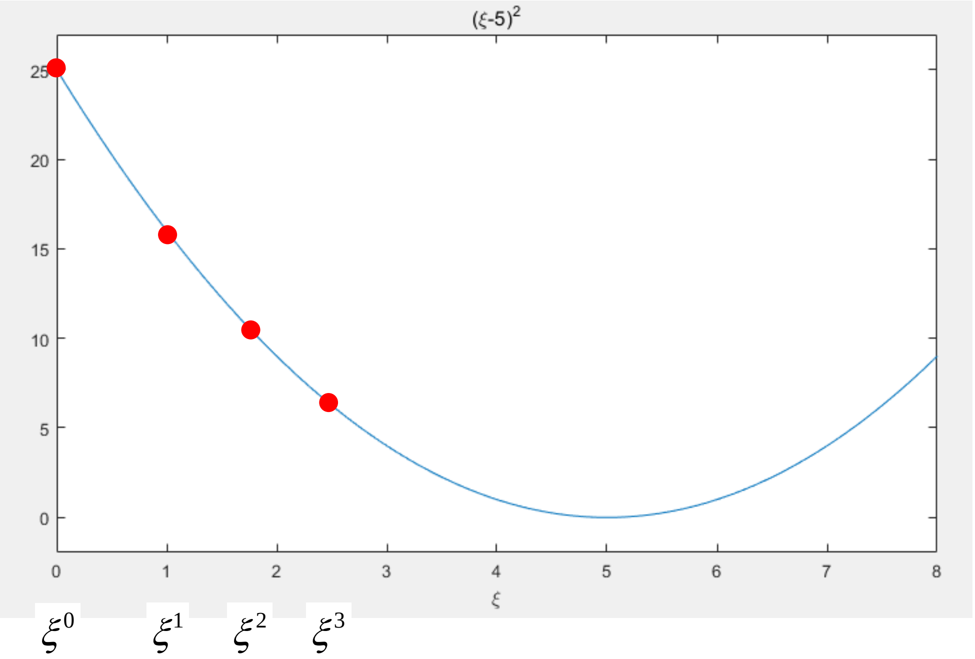
$$\left.\frac{\partial E}{\partial \xi}\right|_{\xi^0} = -10 \qquad \xi^1 = 1$$

$$\xi^1 = 1 \qquad \xi^2 \leftarrow 1 - 0.1(-8)$$

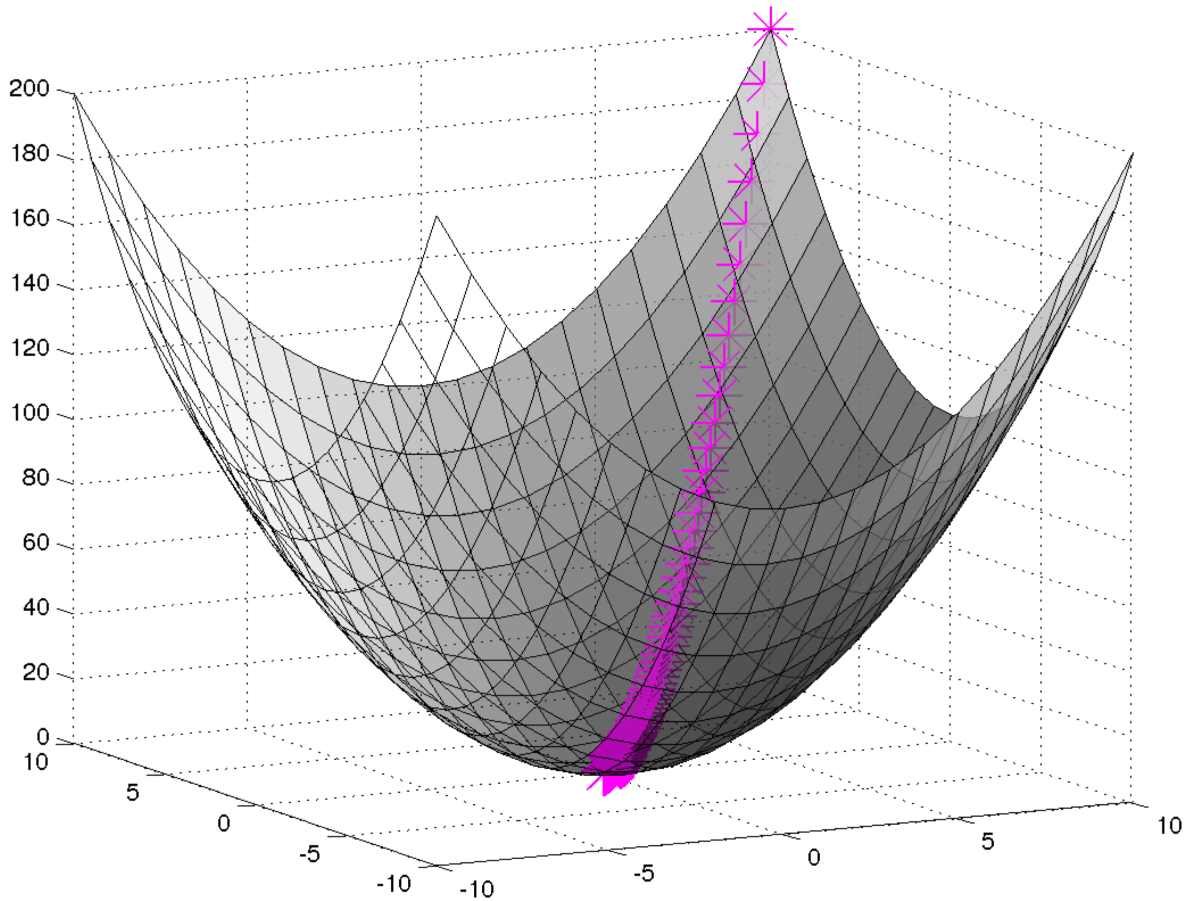
$$\left.\frac{\partial E}{\partial \xi}\right|_{\xi^1} = -8 \qquad \xi^2 = 1.8$$

$$\xi^2 = 1.8 \qquad \xi^3 \leftarrow 1.8 - 0.1(-6.4)$$

$$\left.\frac{\partial E}{\partial \xi}\right|_{\xi^2} = -6.4 \qquad \xi^3 = 2.44$$



Multi Dimension Example 1



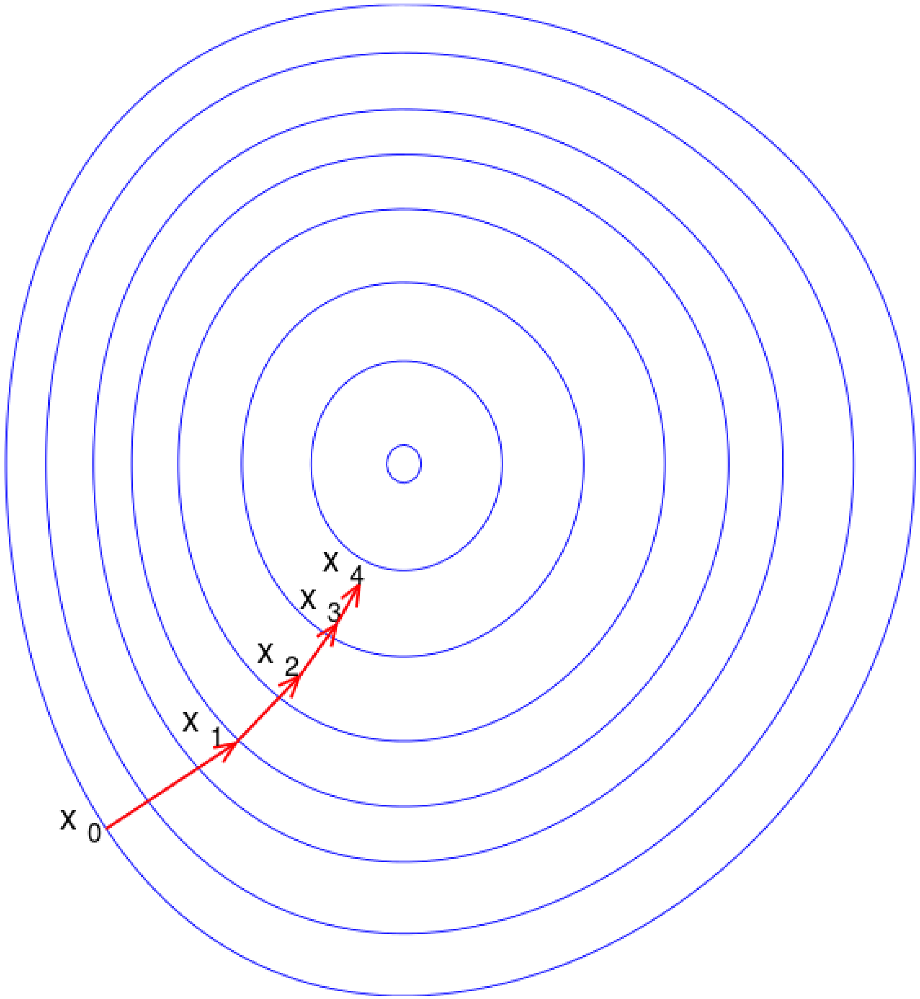
- Gradient vector

$$\nabla J(\underline{\theta}) = \left[\frac{\partial J(\underline{\theta})}{\partial \theta_0} \quad \frac{\partial J(\underline{\theta})}{\partial \theta_1} \quad \ldots \right]$$

- Update rule

$$\underline{\theta}^{k+1} \leftarrow \underline{\theta}^{k} - \alpha \nabla_{\underline{\theta}} J(\underline{\theta}^{k})$$

Multi dimension example 2



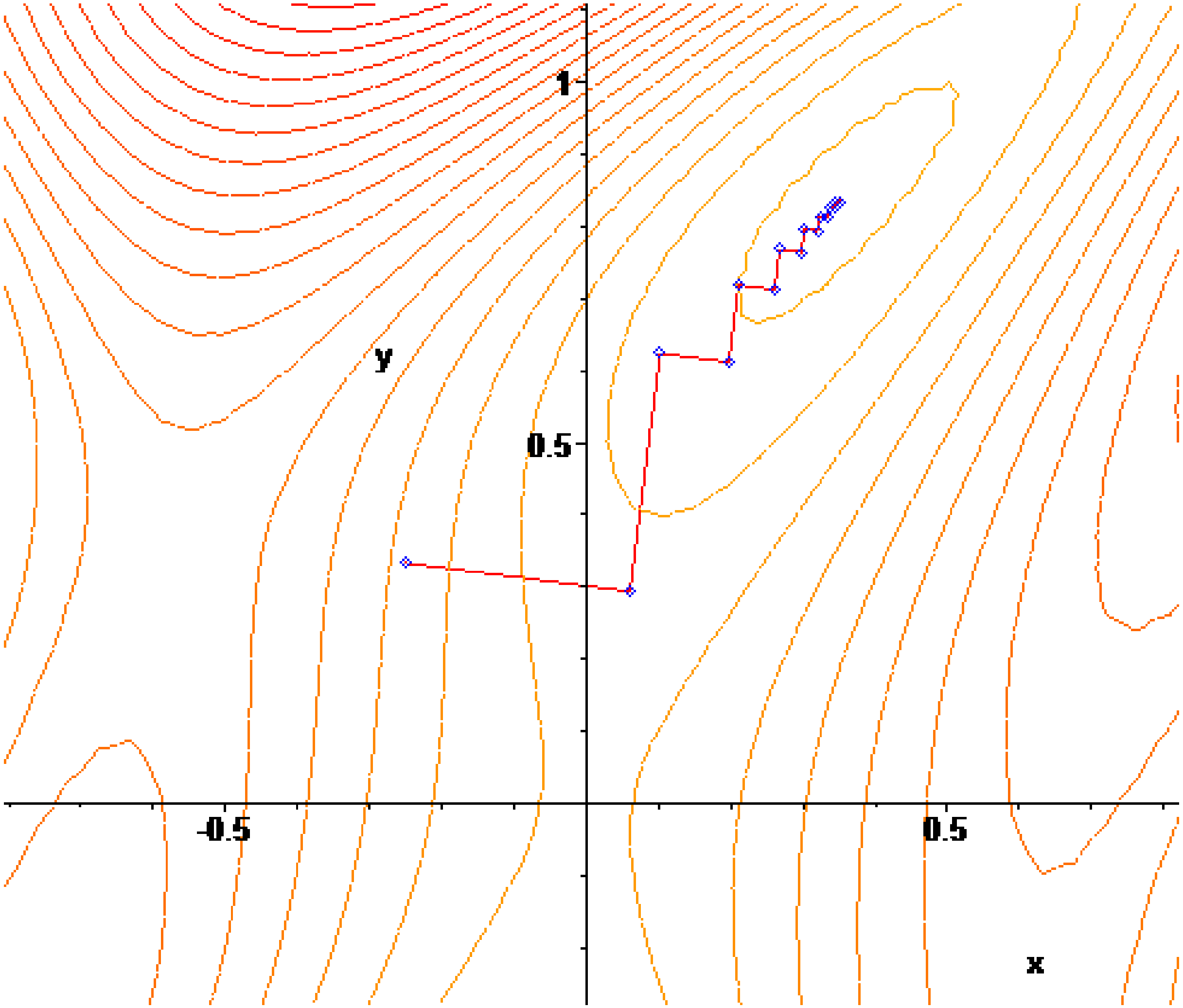
- Gradient vector

$$\nabla J(\underline{\theta}) = \left[\frac{\partial J(\underline{\theta})}{\partial \theta_0} \quad \frac{\partial J(\underline{\theta})}{\partial \theta_1} \quad \ldots \right]$$

- Update rule

$$\underline{\theta}^{k+1} \leftarrow \underline{\theta}^{k} - \alpha \nabla_{\underline{\theta}} J(\underline{\theta}^{k})$$

Multi dimension example 3



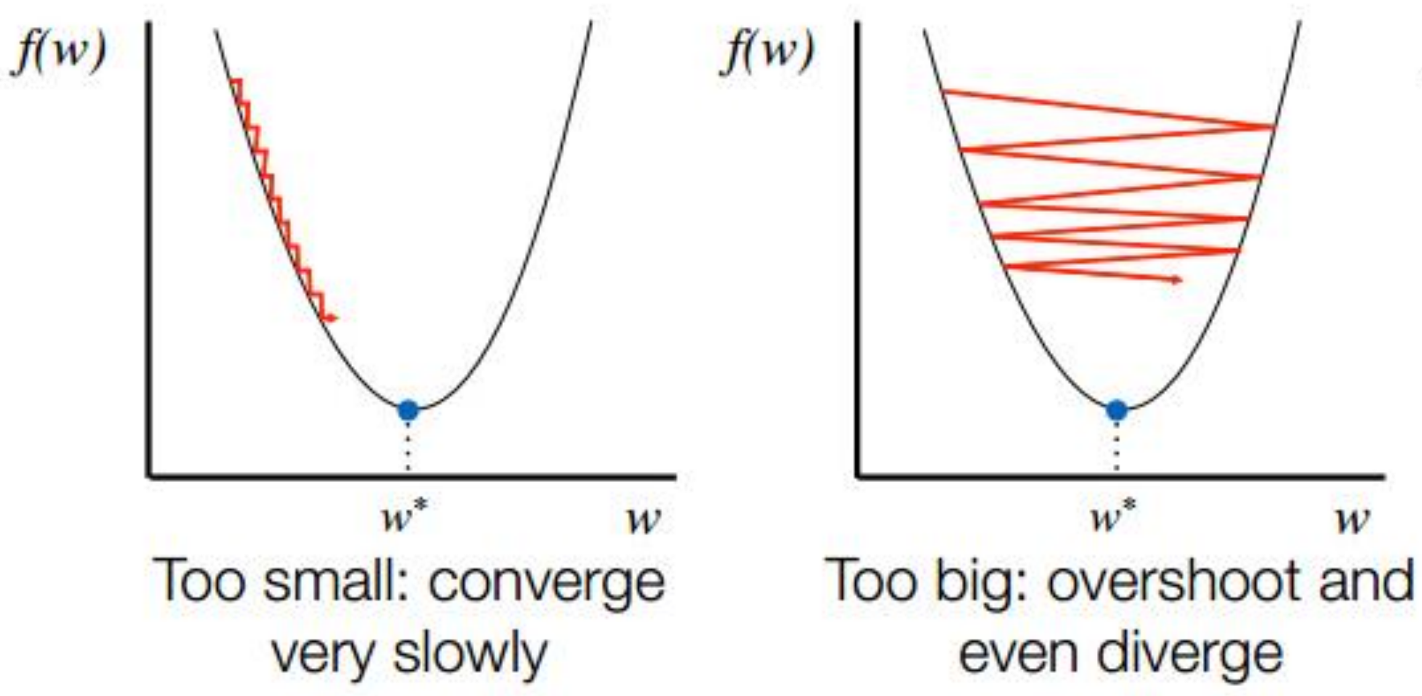
- Gradient vector

$$\nabla J(\underline{\theta}) = \left[\frac{\partial J(\underline{\theta})}{\partial \theta_0} \quad \frac{\partial J(\underline{\theta})}{\partial \theta_1} \quad \cdots \right]$$

- Update rule

$$\underline{\theta}^{k+1} \leftarrow \underline{\theta}^{k} - \alpha \nabla_{\underline{\theta}} J(\underline{\theta}^{k})$$

Gradient descent problem

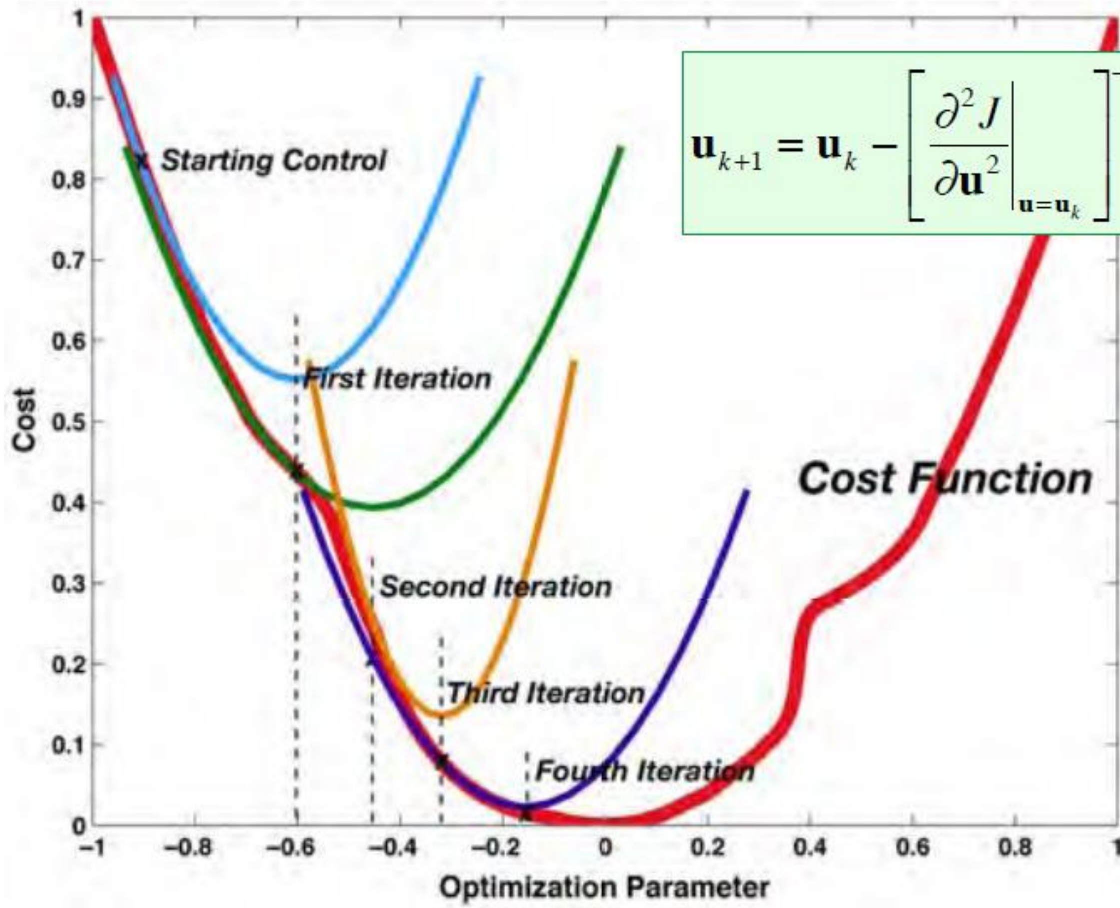


- Update rule

$$\underline{\theta}^{k+1} \leftarrow \underline{\theta}^{k} - \alpha \nabla_{\underline{\theta}} J(\underline{\theta}^{k})$$

Newton-Raphson Iteration

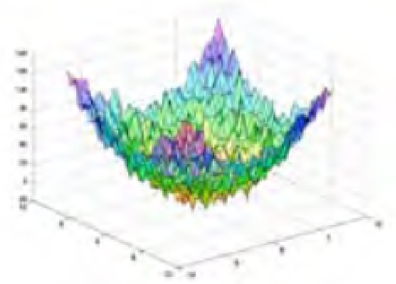
Newton-Raphson algorithm is an iterative search using both the gradient and the Hessian matrix



$$\mathbf{u}_{k+1} = \mathbf{u}_k - \left[\left. \frac{\partial^2 J}{\partial \mathbf{u}^2} \right|_{\mathbf{u}=\mathbf{u}_k} \right]^{-1} \left[\left. \frac{\partial J}{\partial \mathbf{u}} \right|_{\mathbf{u}=\mathbf{u}_k} \right]^T$$

$$\underline{\theta}^{k+1} \leftarrow \underline{\theta}^{k} - \alpha \nabla_{\underline{\theta}} J(\underline{\theta}^{k})$$

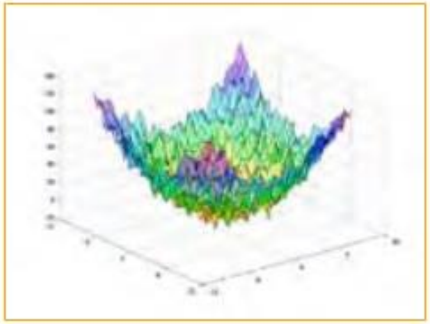
$$\underline{\theta}^{k+1} \leftarrow \underline{\theta}^{k} - \nabla^2_{\theta} J(\underline{\theta}^{k})^{-1} \nabla_{\underline{\theta}} J(\underline{\theta}^{k})$$



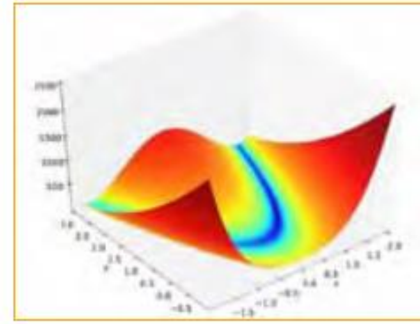
Difficulties with Newton-Raphson Iteration

$$\mathbf{u}_{k+1} = \mathbf{u}_k - \boxed{\left[\left. \frac{\partial^2 J}{\partial \mathbf{u}^2} \right|_{\mathbf{u}=\mathbf{u}_k} \right]}^{-1} \left[\left. \frac{\partial J}{\partial \mathbf{u}} \right|_{\mathbf{u}=\mathbf{u}_k} \right]^T$$

- Good when close to the optimum, but …
- Hessian matrix (i.e., the curvature) may be
  - Hard to estimate, e.g., large effects of small errors
  - Locally misleading, e.g., wrong curvature
- Gradient searches focus on local minima



Gradient Search Issues



Steepest Descent

$$\mathbf{u}_{k+1} = \mathbf{u}_k - \varepsilon \left[\left. \frac{\partial J}{\partial \mathbf{u}} \right|_{\mathbf{u}=\mathbf{u}_k} \right]^T$$

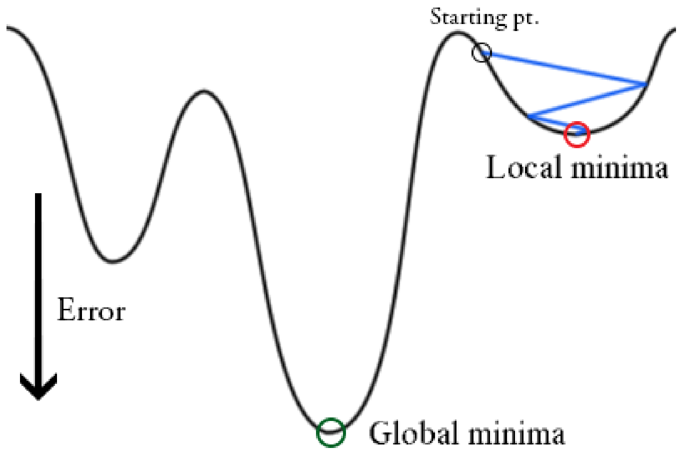
Newton Raphson

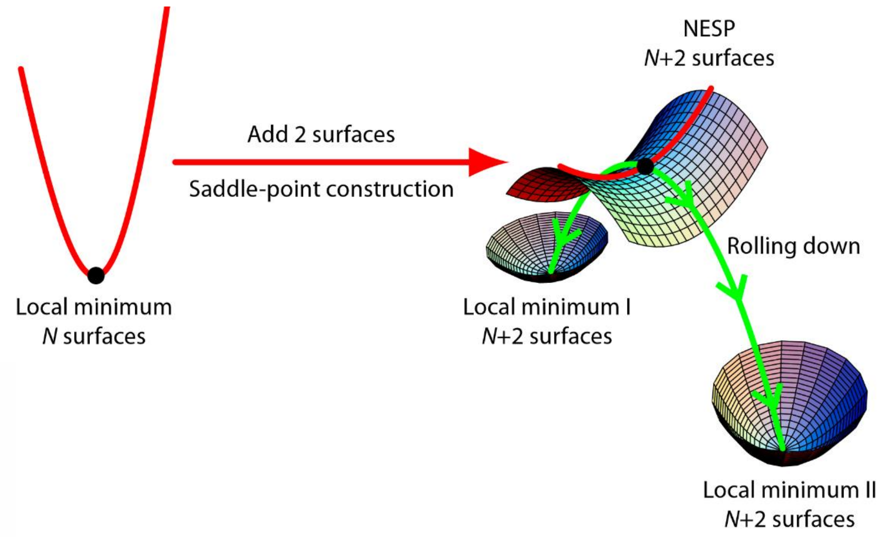
$$\mathbf{u}_{k+1} = \mathbf{u}_k - \left[\left. \frac{\partial^2 J}{\partial \mathbf{u}^2} \right|_{\mathbf{u}=\mathbf{u}_k} \right]^{-1} \left[\left. \frac{\partial J}{\partial \mathbf{u}} \right|_{\mathbf{u}=\mathbf{u}_k} \right]^T$$

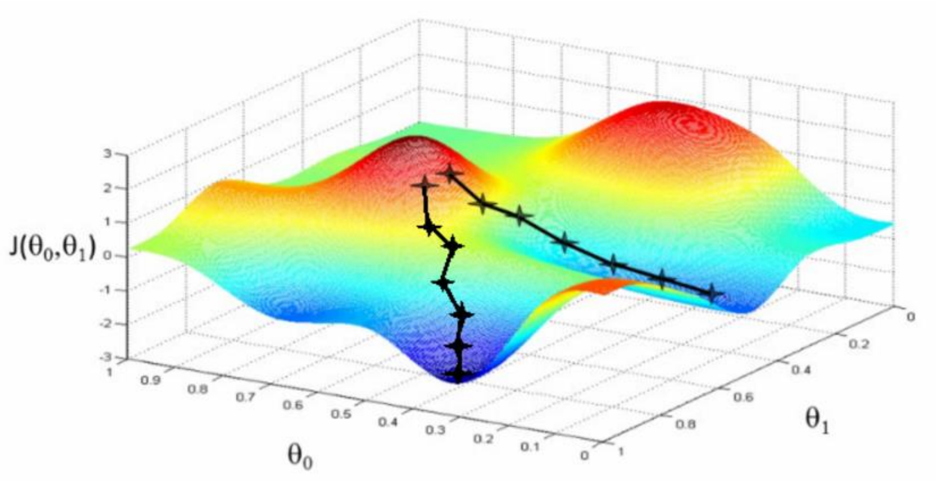
- Need to evaluate gradient (and possibly Hessian matrix)
- Not global: gradient searches focus on local minima
- Convergence may be difficult with "noisy" or complex cost functions

13

Local Minima Problem







Two Types Gradient-Free Search

- Find Global Minima

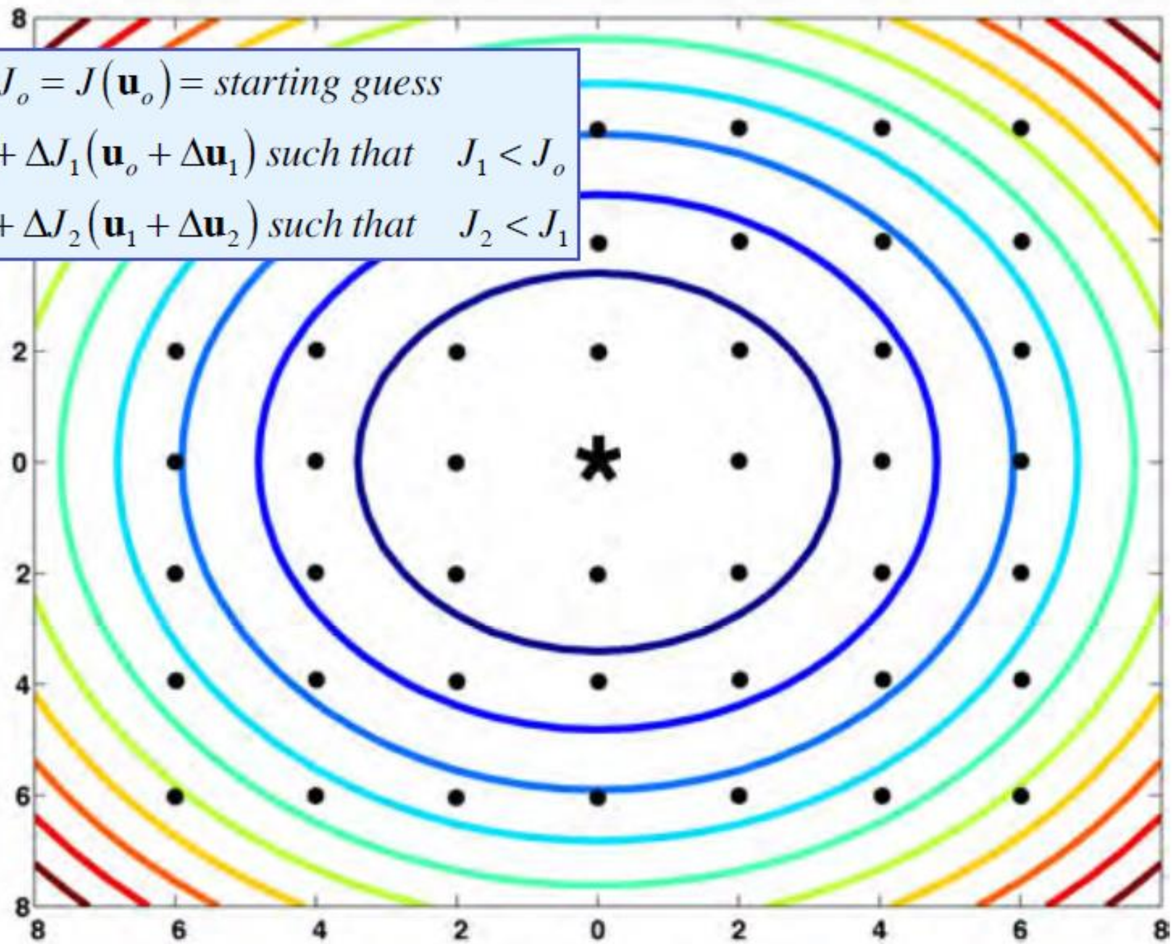
- Just Gradient-Free

Gradient-Free Search:
Grid-Based Search

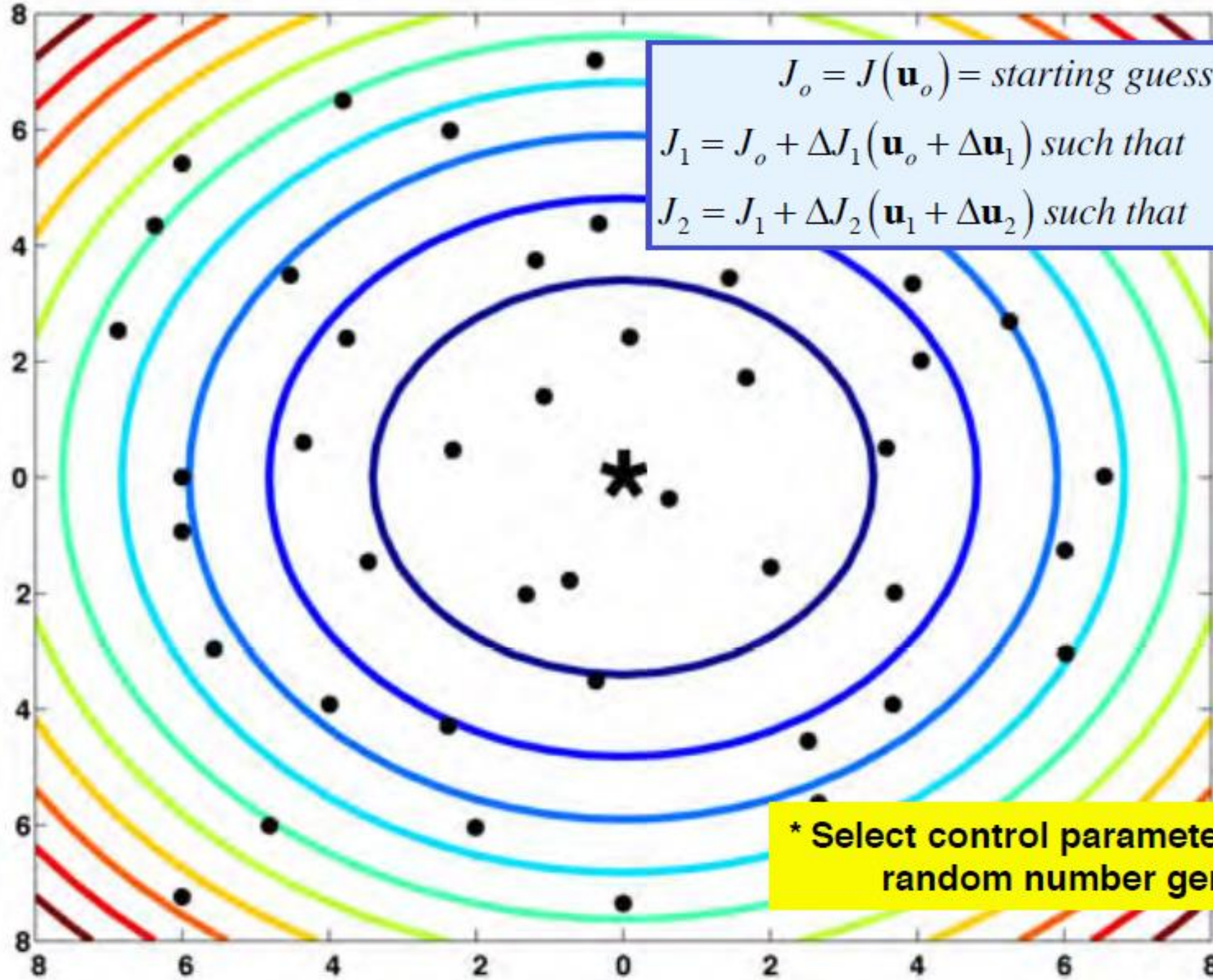
$$J_o = J(\mathbf{u}_o) = \textit{starting guess}$$

$$J_1 = J_o + \Delta J_1(\mathbf{u}_o + \Delta\mathbf{u}_1) \textit{ such that } \quad J_1 < J_o$$

$$J_2 = J_1 + \Delta J_2(\mathbf{u}_1 + \Delta\mathbf{u}_2) \textit{ such that } \quad J_2 < J_1$$



Gradient-Free Search:
Random Search



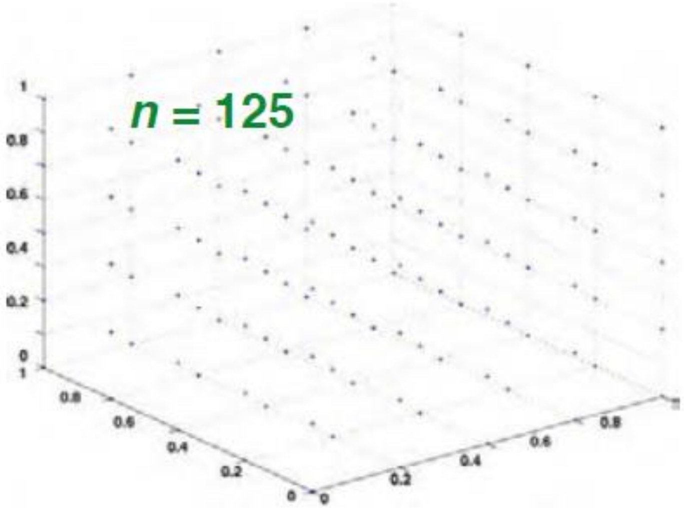
$$J_o = J(\mathbf{u}_o) = \textit{starting guess}$$

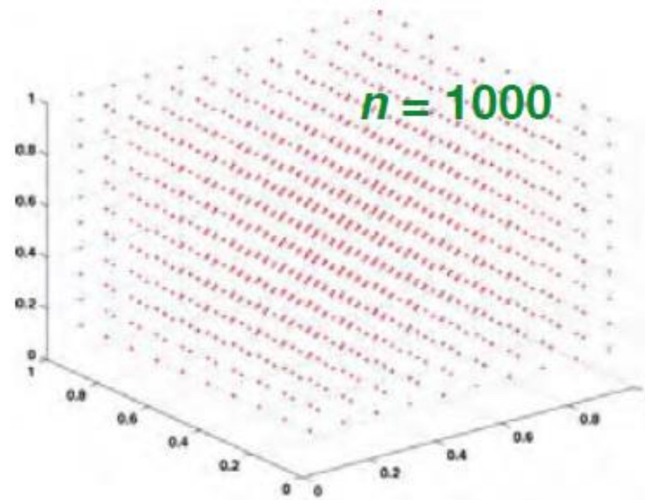
$$J_1 = J_o + \Delta J_1(\mathbf{u}_o + \Delta\mathbf{u}_1) \textit{ such that } \quad J_1 < J_o$$

$$J_2 = J_1 + \Delta J_2(\mathbf{u}_1 + \Delta\mathbf{u}_2) \textit{ such that } \quad J_2 < J_1$$

*** Select control parameters using a random number generator**

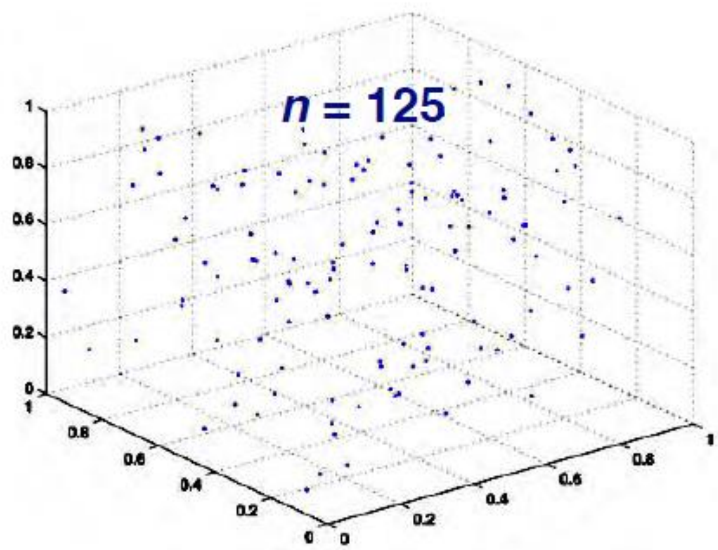
Three-Parameter Grid Search

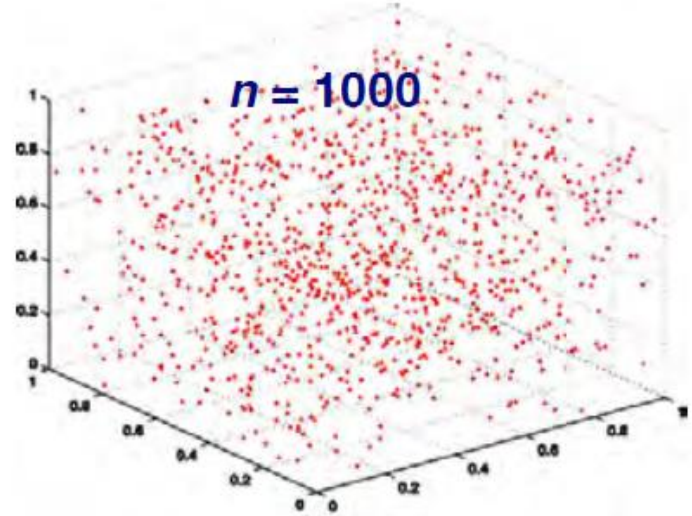




- **Regular spacing**
- **Fixed resolution**
- **Trials grow as m^n, where**
  - n = Number of parameters
  - m = Resolution

Three-Parameter Random Field Search





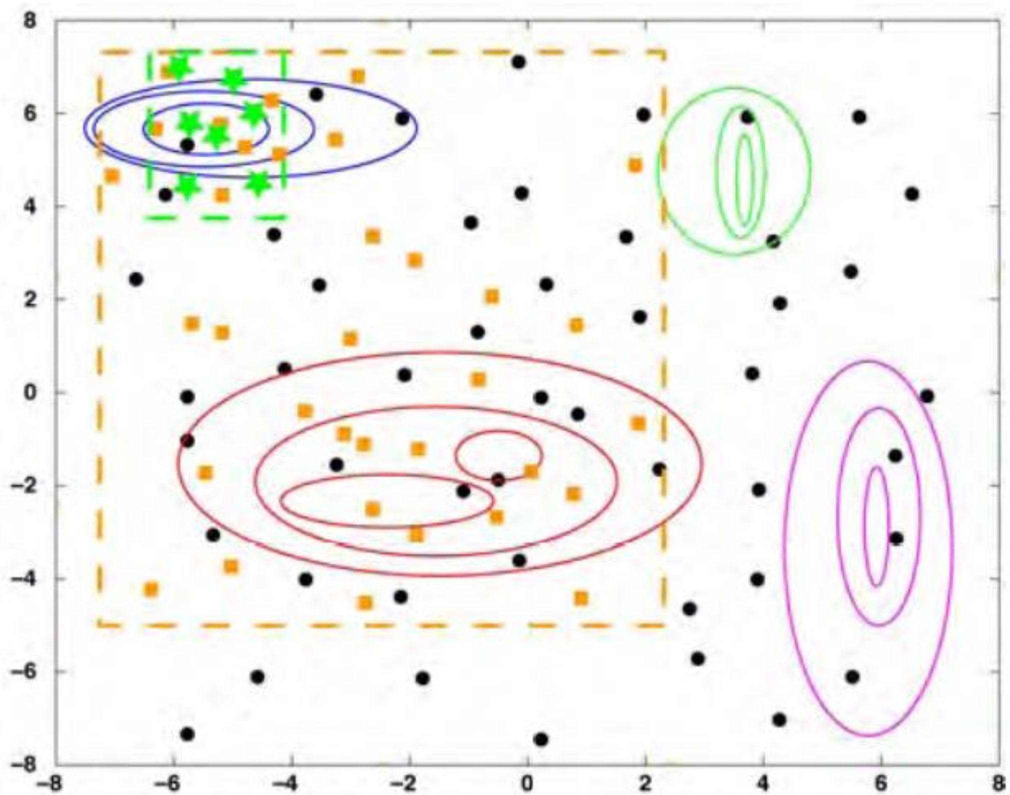
Variable spacing and resolution
Arbitrary number of trials
Random space-filling

Directed (Structured) Search for Minimum Cost

Continuation of grid-based or random search

Localize areas of low cost

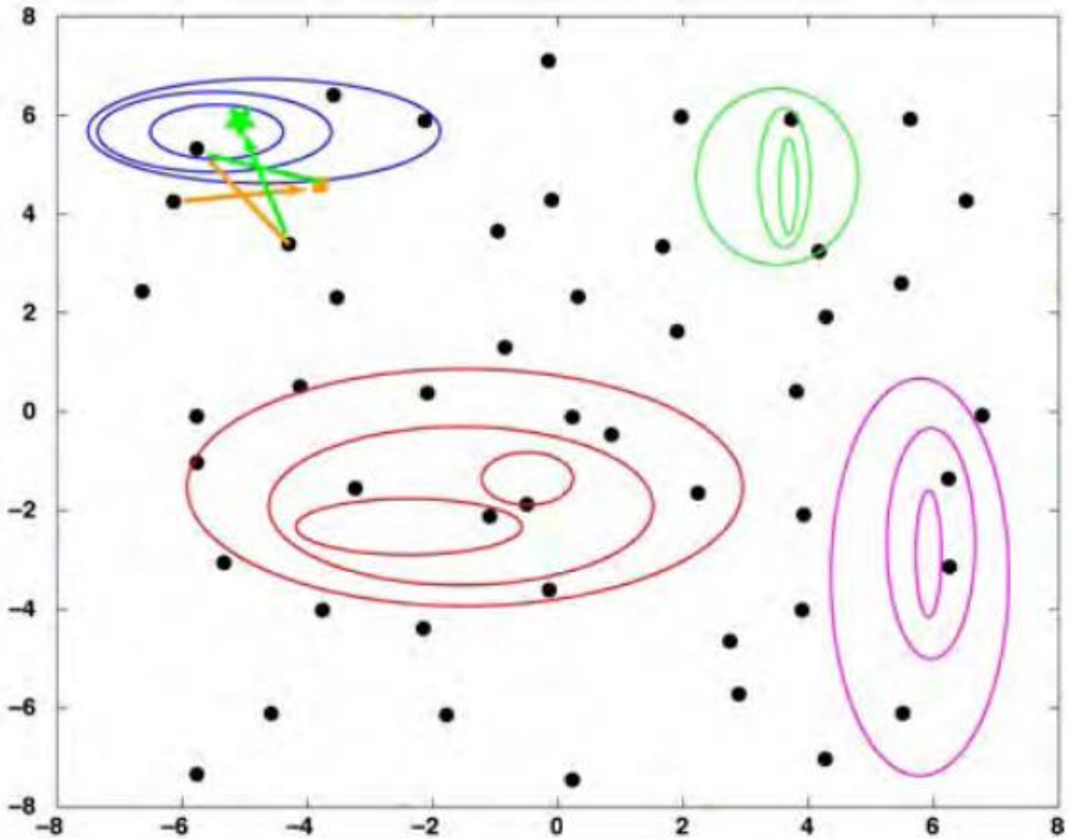
Increase sampling density in those areas



18

Directed (Structured) Search for Minimum Cost

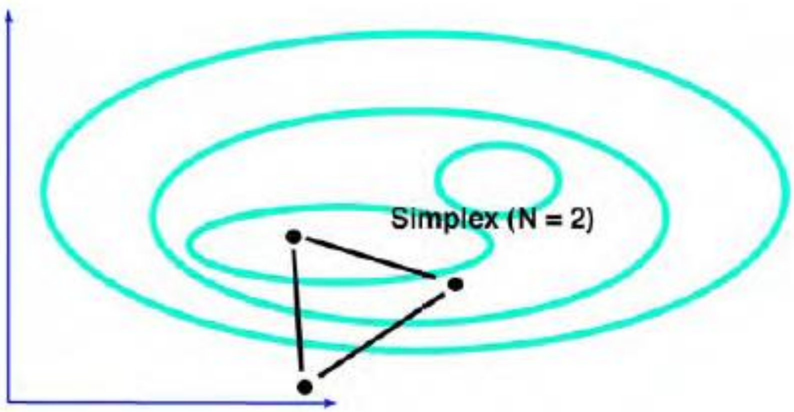
- Interpolate or extrapolate from one or more starting points

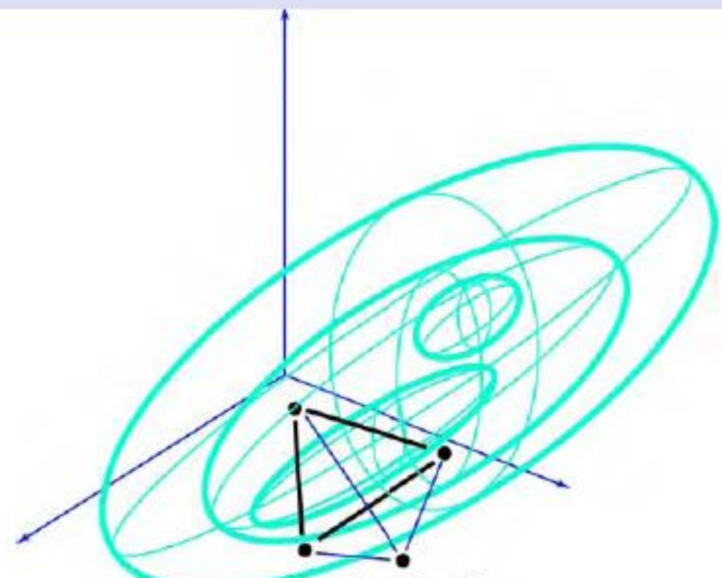


19

Downhill Simplex Search (Nelder-Mead Algorithm)

- Simplex: N-dimensional figure in control space defined by
  - $N+1$ vertices
  - $(N+1)\,N/2$ straight edges between vertices

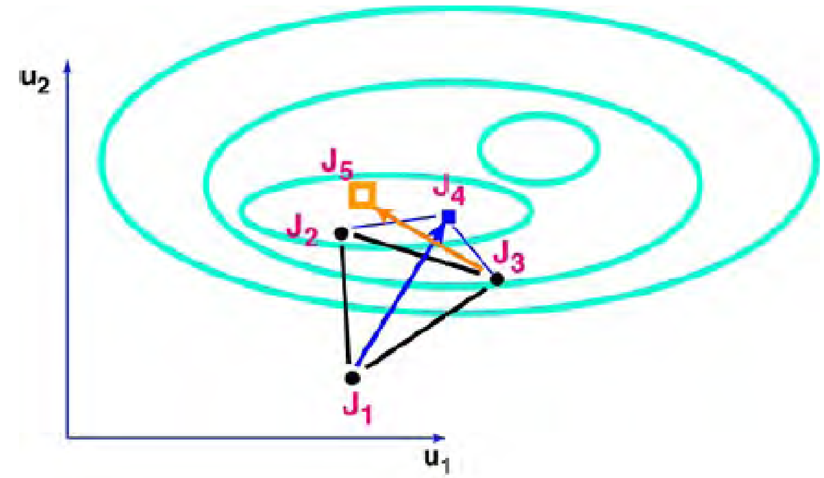




20

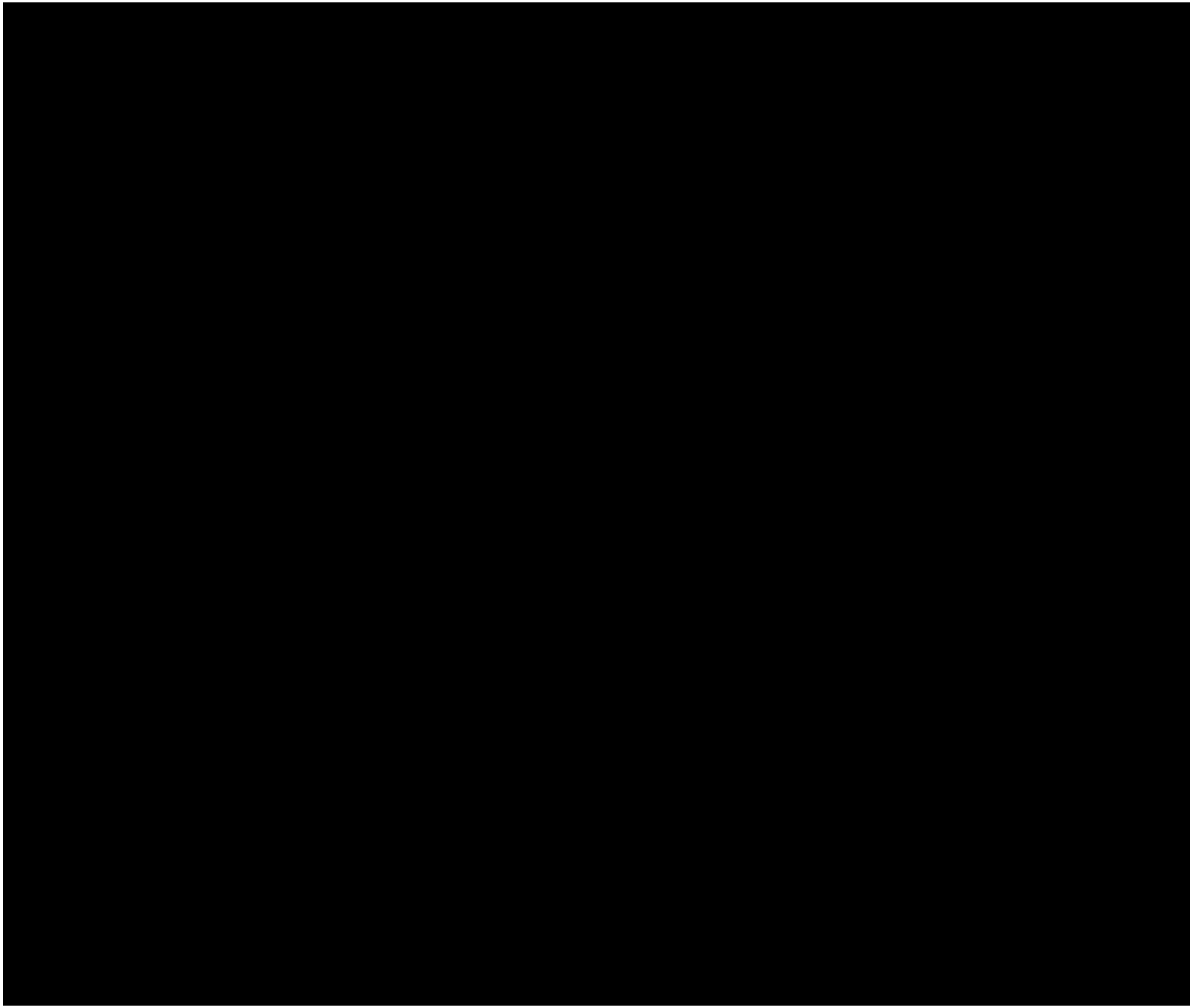
Search Procedure for Downhill Simplex Method

- Select starting set of vertices
- Evaluate cost at each vertex
- Determine vertex with largest cost (e.g., J_1 at right)



- Project search from this vertex through middle of opposite face (or edge for $N = 2$)
- Evaluate cost at new vertex (e.g., J_4 at right)
- Drop J_1 vertex, and form simplex with new vertex
- Repeat until cost is small

Humanoid Walker optimized via Nelder-Mead
http://www.youtube.com/watch?v=BcYPLR_j5dg



Tutorial

- Frame to frame motion estimation (16.11.14)
- Numerical optimization (16.12.12)
- Graph SLAM
- Loop closure detection

Paper study

- Feature based method

  1. Real-time Depth Enhanced Monocular Odometry

  2. Lidar Odometry and Mapping in Real-time

- Direct method

  3. LSD-SLAM

  4. Large-Scale Direct SLAM with Stereo Cameras

  5. Semi-Direct Visual Odometry for a fisheye-stereo camera