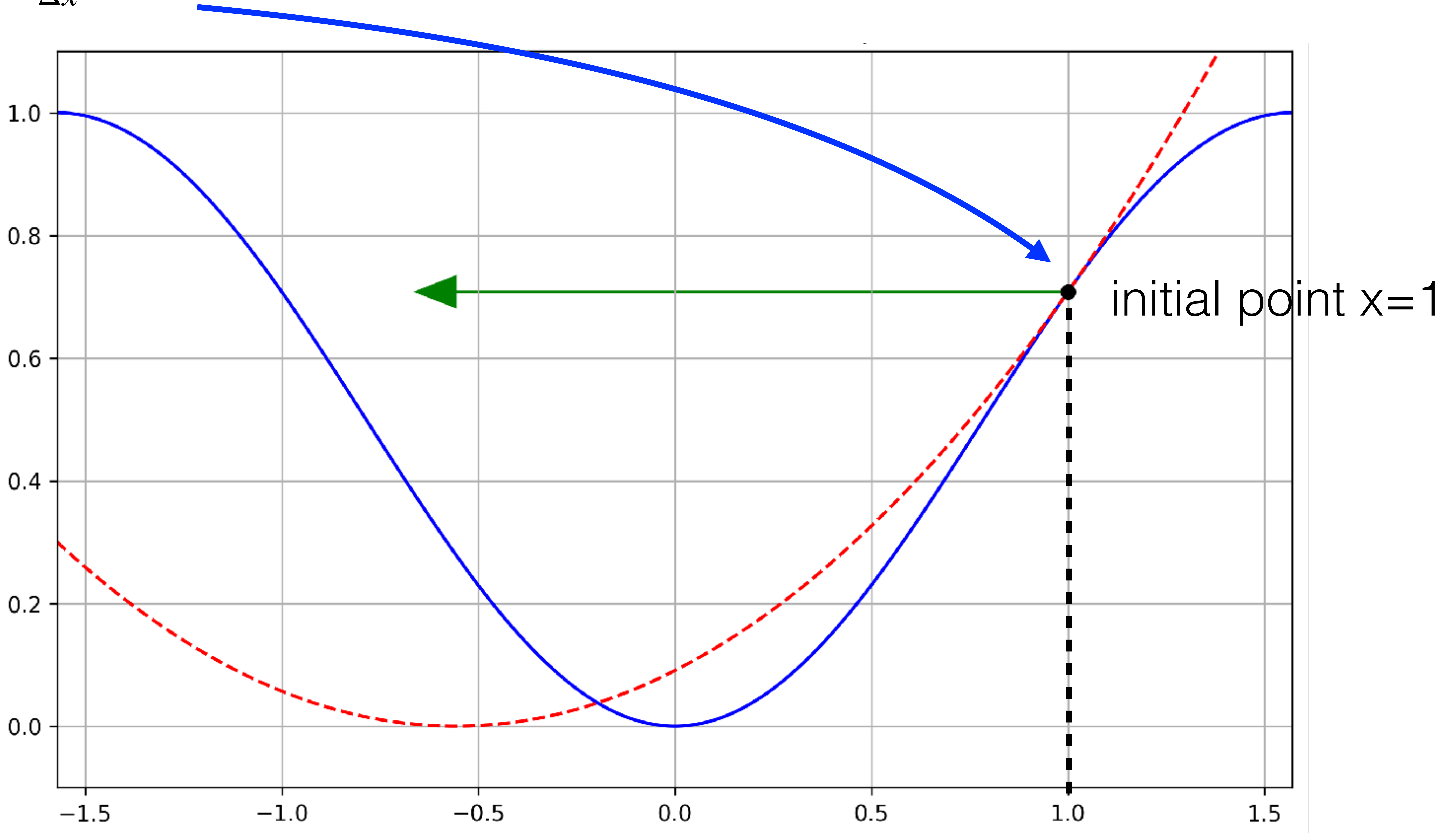
Factorgraph Optimization

Karel Zimmermann

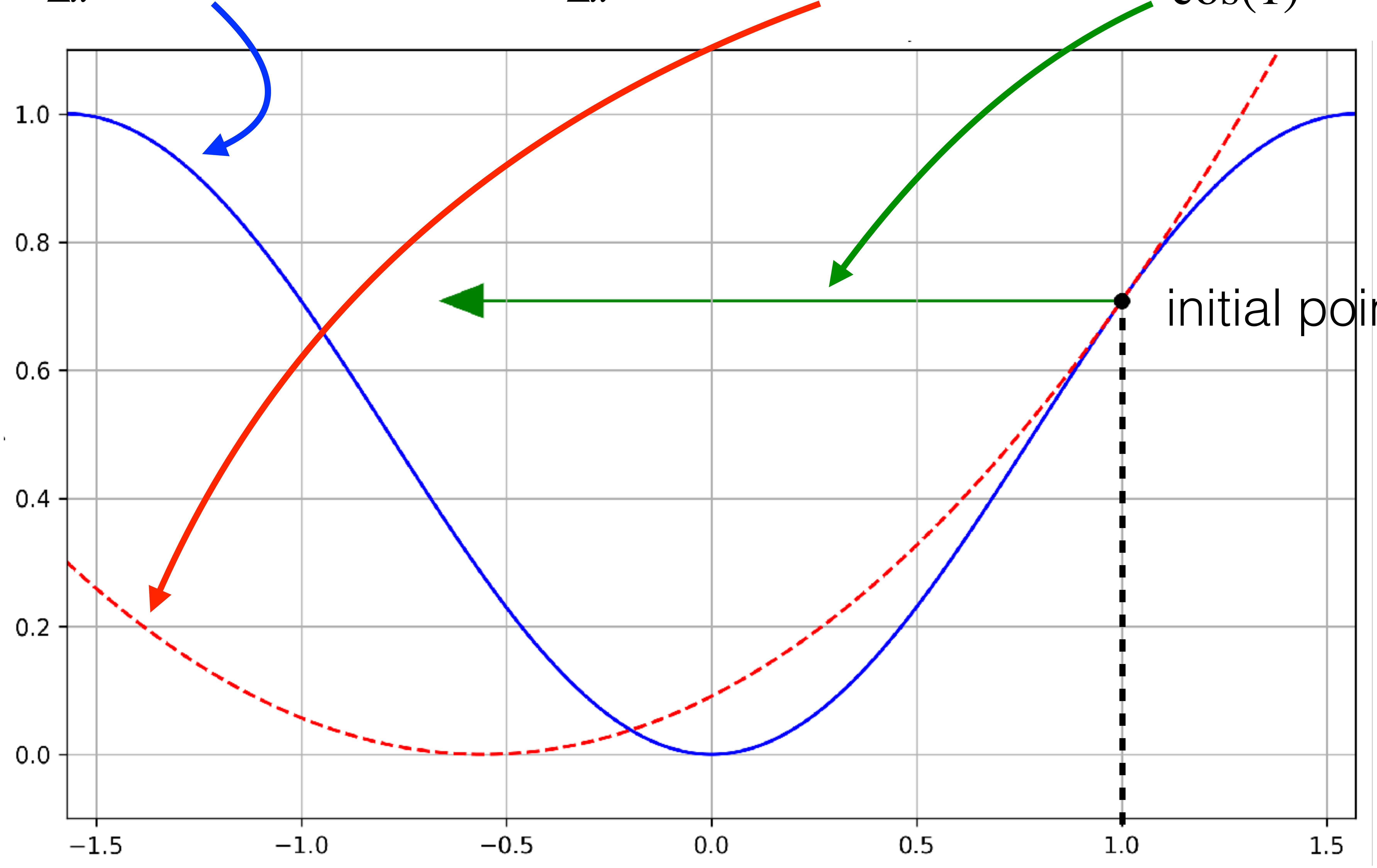
Prerequisites: Gauss-Newton method

$$\Delta x^* = \arg\min_{\Delta x} \left[\sin(x + \Delta x)\right]^2$$



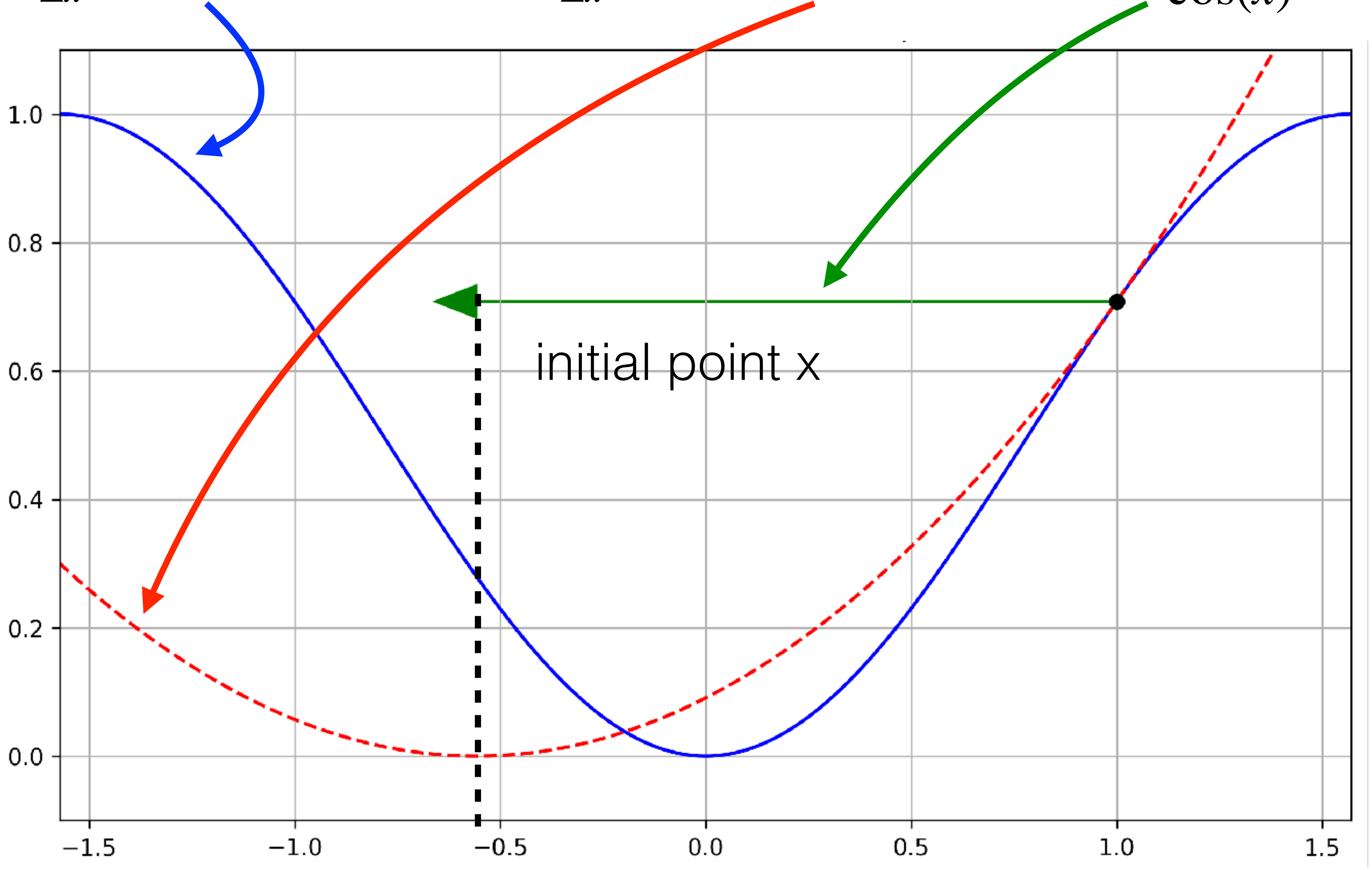
Prerequisites: Gauss-Newton method

$$\Delta x^* = \arg\min_{\Delta x} \left[\sin(1+\Delta x)\right]^2 \approx \arg\min_{\Delta x} \left[\sin(1) + \cos(1)\Delta x\right]^2 = -\frac{\sin(1)}{\cos(1)} = -\tan(1)$$



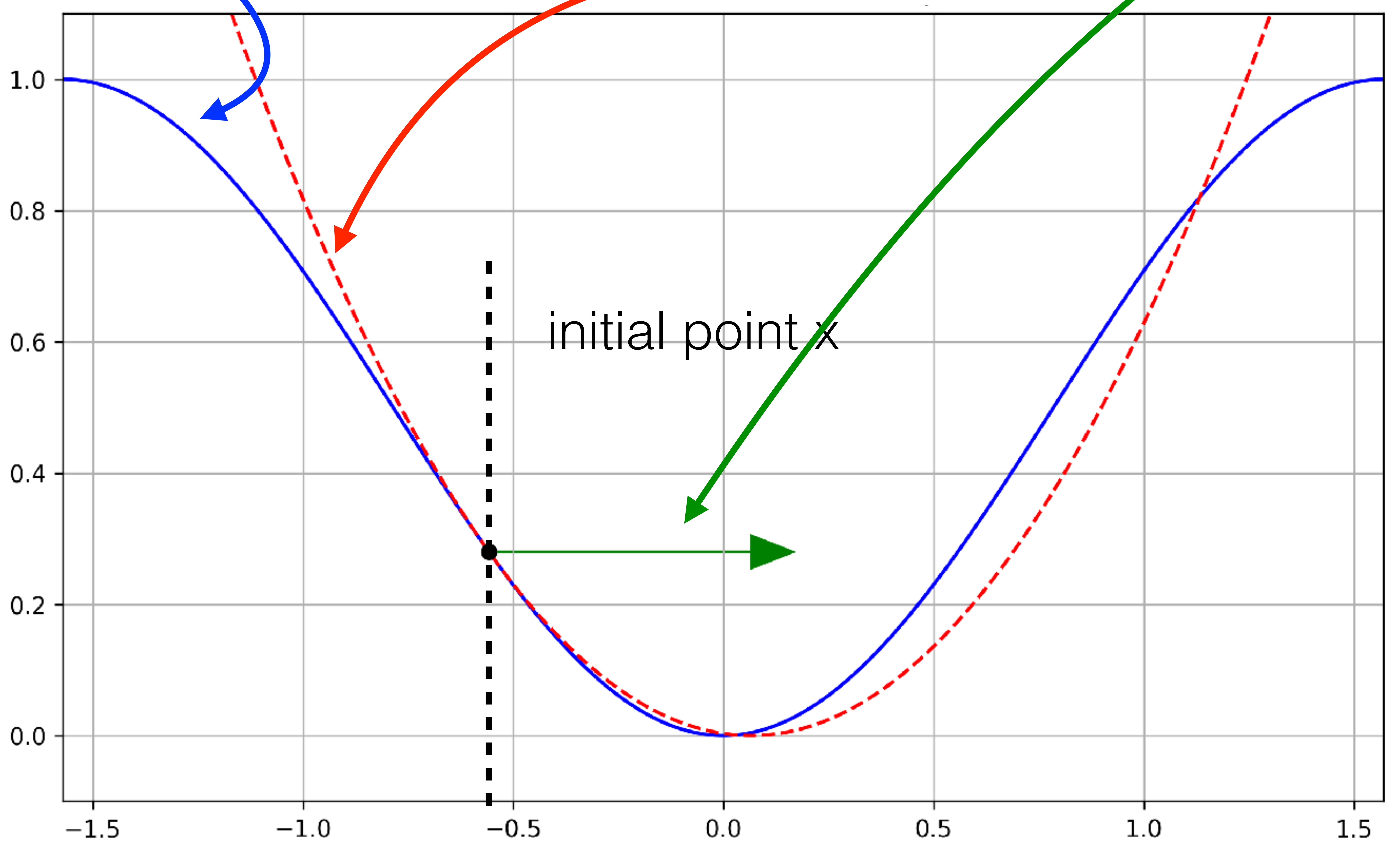
Prerequisites: Gauss-Newton method

$$\Delta x^* = \arg\min_{\Delta x} \left[\sin(x + \Delta x)\right]^2 \approx \arg\min_{\Delta x} \left[\sin(x) + \cos(x)\Delta x\right]^2 = -\frac{\sin(x)}{\cos(x)} = -\tan(x)$$



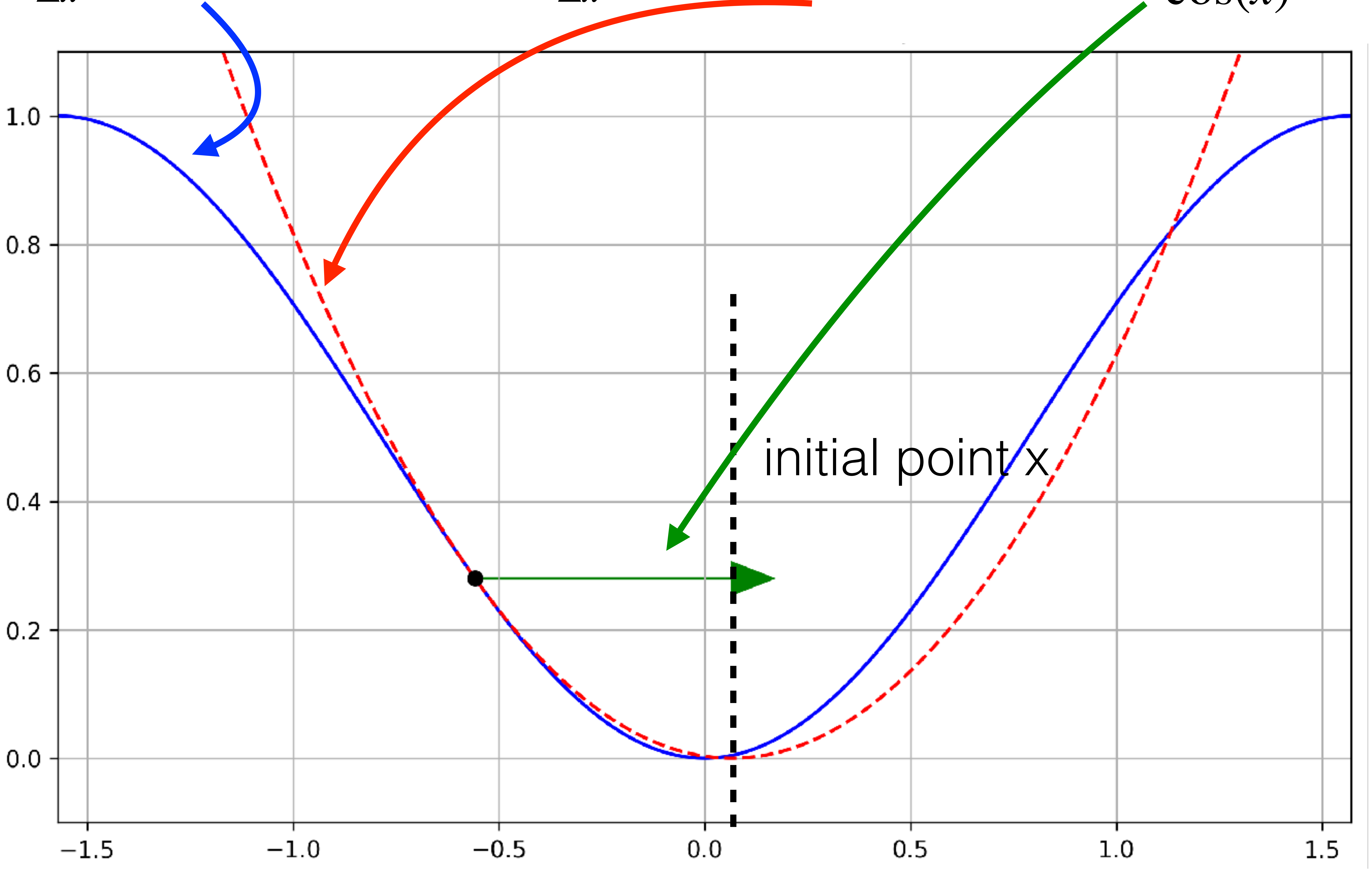
Prerequisites: Gauss-Newton method

$$\Delta x^* = \arg\min_{\Delta x} \left[\sin(x + \Delta x)\right]^2 \approx \arg\min_{\Delta x} \left[\sin(x) + \cos(x)\Delta x\right]^2 = -\frac{\sin(x)}{\cos(x)} = -\tan(x)$$



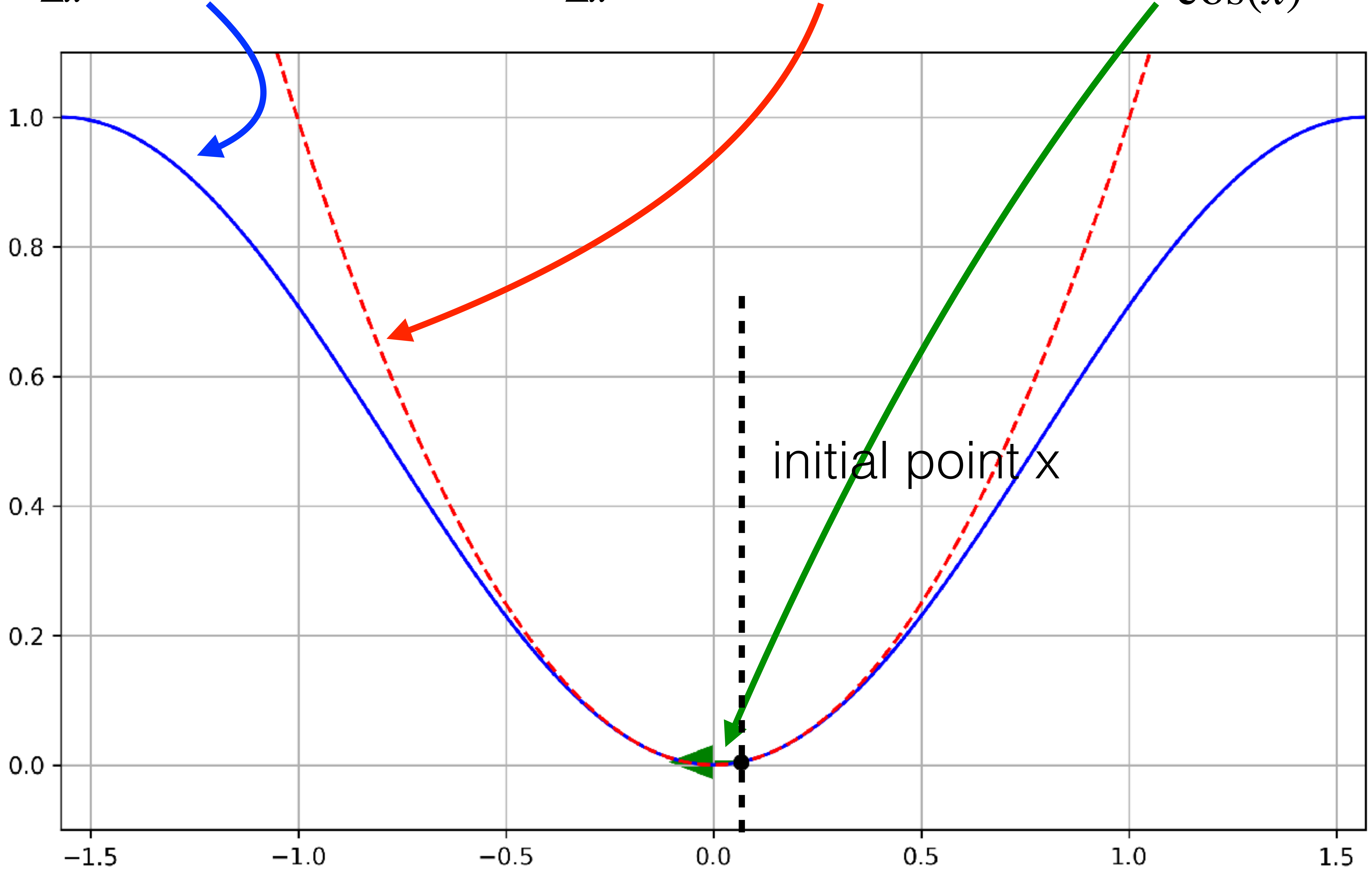
Prerequisites: Gauss-Newton method

$$\Delta x^* = \arg\min_{\Delta x} \left[\sin(x + \Delta x)\right]^2 \approx \arg\min_{\Delta x} \left[\sin(x) + \cos(x)\Delta x\right]^2 = -\frac{\sin(x)}{\cos(x)} = -\tan(x)$$



Prerequisites: Gauss-Newton method

$$\Delta x^* = \arg\min_{\Delta x} \left[\sin(x+\Delta x)\right]^2 \approx \arg\min_{\Delta x} \left[\sin(x) + \cos(x)\Delta x\right]^2 = -\frac{\sin(x)}{\cos(x)} = -\tan(x)$$



Prerequisites: Gauss-Newton method

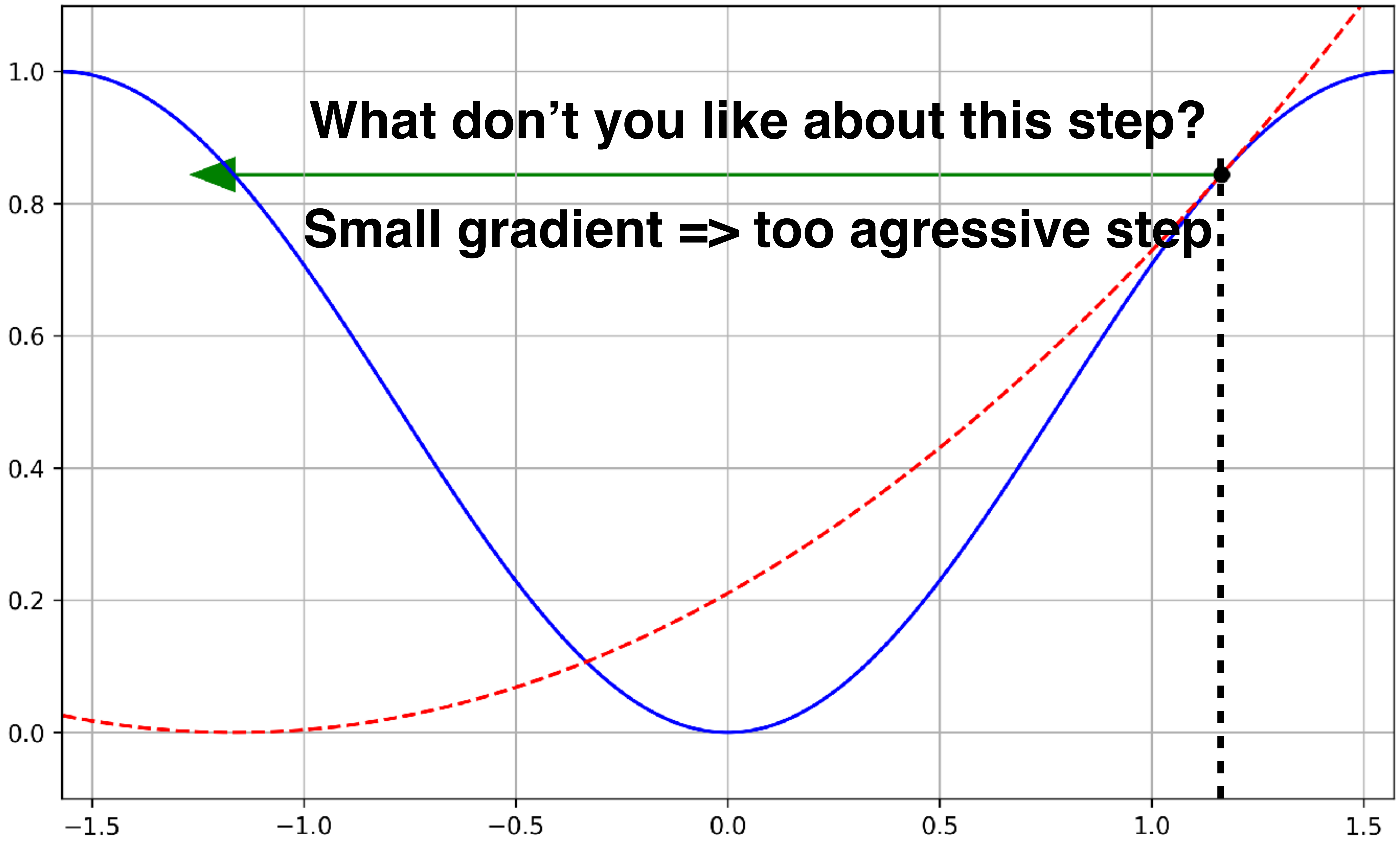
$$\Delta x^* = \arg\min_{\Delta x} \left[\sin(x + \Delta x)\right]^2 \approx \arg\min_{\Delta x} \left[\sin(x) + \cos(x)\Delta x\right]^2 = -\frac{\sin(x)}{\cos(x)} = -\tan(x)$$

$$\Delta x^* = \arg\min_{\Delta x} \left[f(x + \Delta x)\right]^2 \quad \approx \arg\min_{\Delta x} \left[f(x) + f'(x)\Delta x\right]^2$$

$$= \arg\min_{\Delta x} \; \underbrace{f'(x)^2}_{\text{Hessian}}\Delta x^2 + 2f'(x)\Delta x + f(x)^2 \quad = -\frac{f(x)}{f'(x)}$$

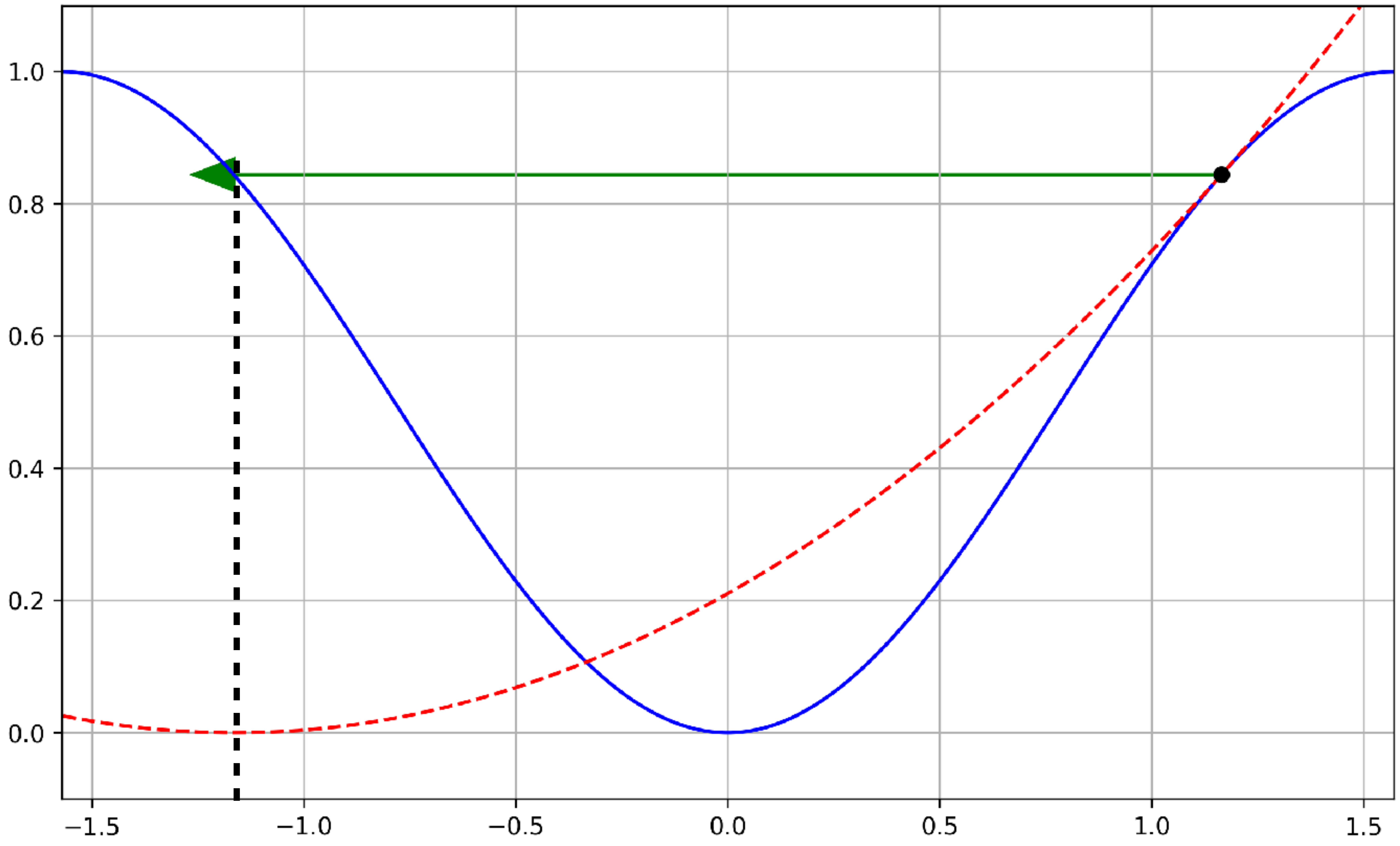
Prerequisites: Gauss-Newton method

$$\Delta x^* = \arg\min_{\Delta x} \left[\sin(1+\Delta x)\right]^2 \approx \arg\min_{\Delta x} \left[\sin(1) + \cos(1)\Delta x\right]^2 = -\frac{\sin(1)}{\cos(1)} = -\tan(1)$$



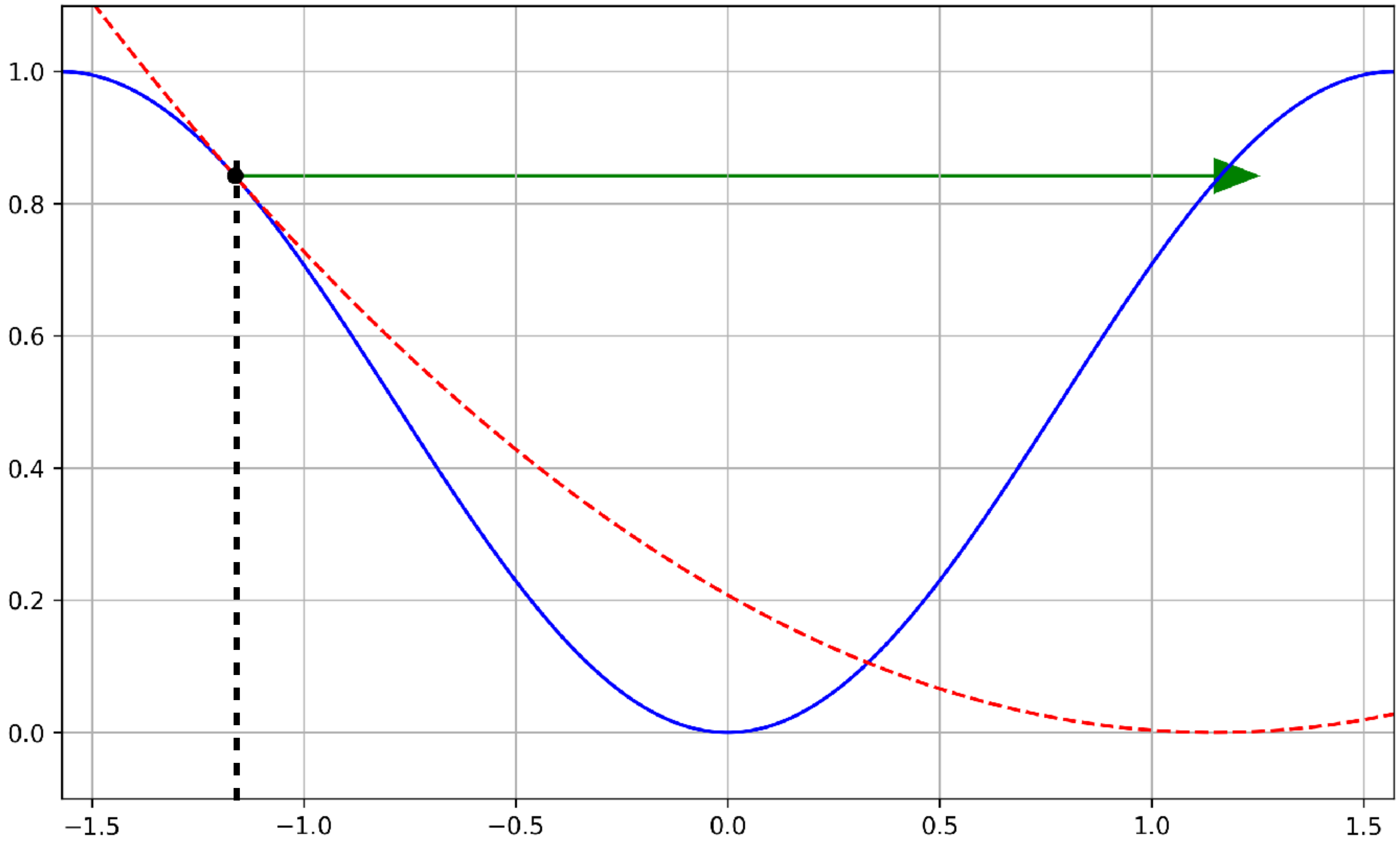
Prerequisites: Gauss-Newton method

$$\Delta x^* = \arg\min_{\Delta x} \left[\sin(1+\Delta x)\right]^2 \approx \arg\min_{\Delta x} \left[\sin(1) + \cos(1)\Delta x\right]^2 = -\frac{\sin(1)}{\cos(1)} = -\tan(1)$$



Prerequisites: Gauss-Newton method

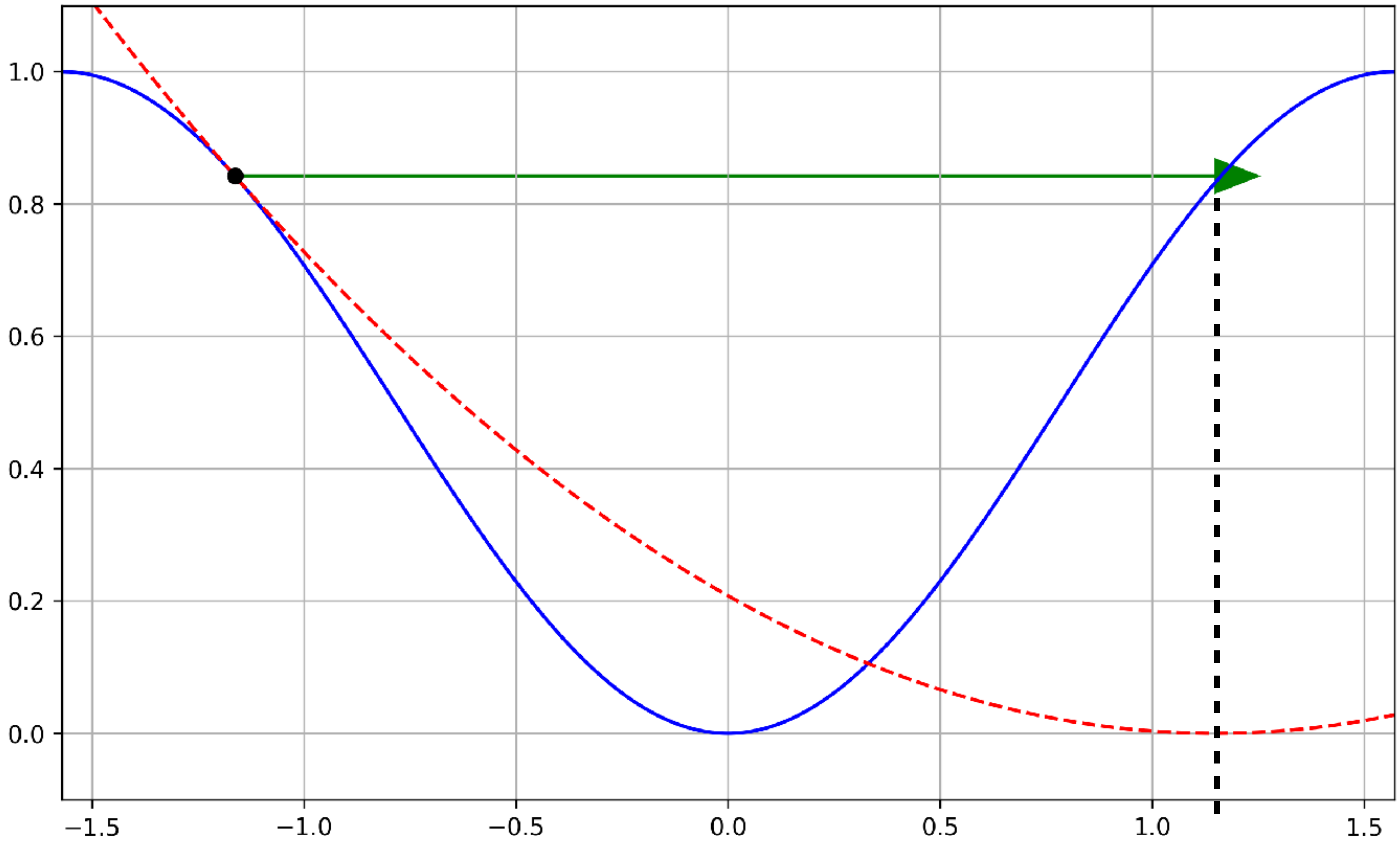
$$\Delta x^* = \arg\min_{\Delta x} \left[\sin(1+\Delta x)\right]^2 \approx \arg\min_{\Delta x} \left[\sin(1) + \cos(1)\Delta x\right]^2 = -\frac{\sin(1)}{\cos(1)} = -\tan(1)$$



ɔint x=1

Prerequisites: Gauss-Newton method

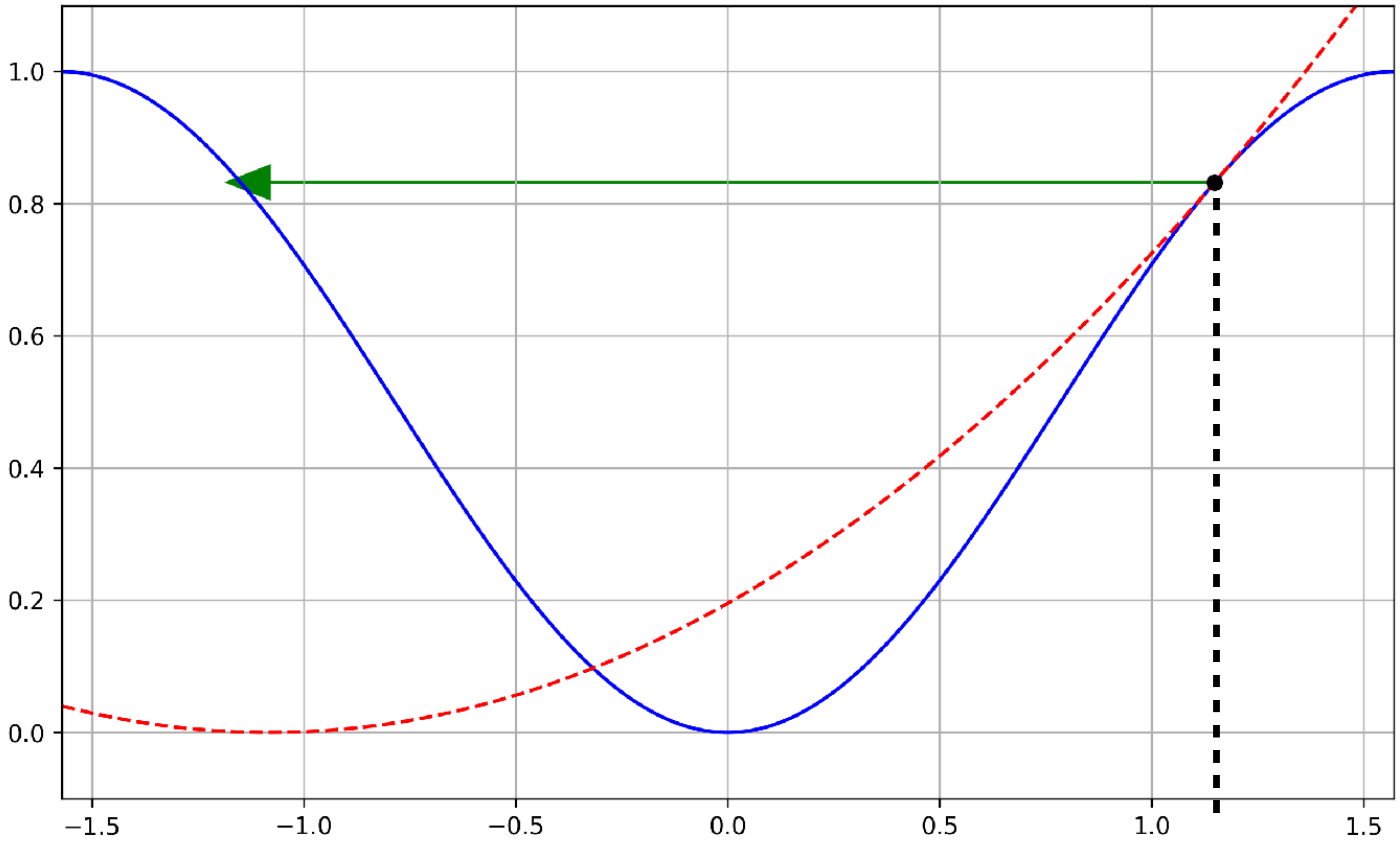
$$\Delta x^* = \arg\min_{\Delta x} \left[\sin(1+\Delta x)\right]^2 \approx \arg\min_{\Delta x} \left[\sin(1) + \cos(1)\Delta x\right]^2 = -\frac{\sin(1)}{\cos(1)} = -\tan(1)$$



›int x=1

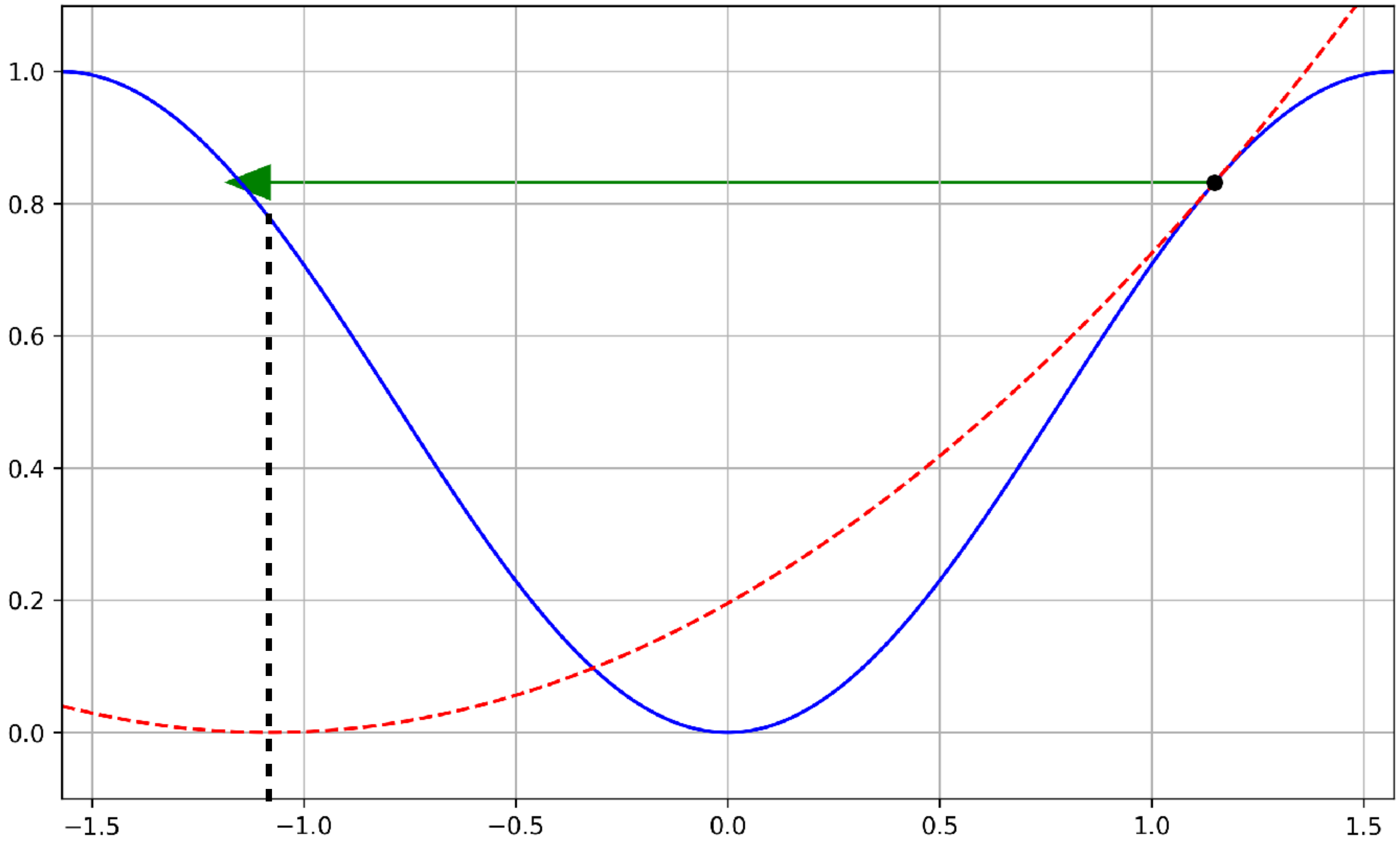
Prerequisites: Gauss-Newton method

$$\Delta x^* = \arg\min_{\Delta x} \left[\sin(1+\Delta x)\right]^2 \approx \arg\min_{\Delta x} \left[\sin(1) + \cos(1)\Delta x\right]^2 = -\frac{\sin(1)}{\cos(1)} = -\tan(1)$$



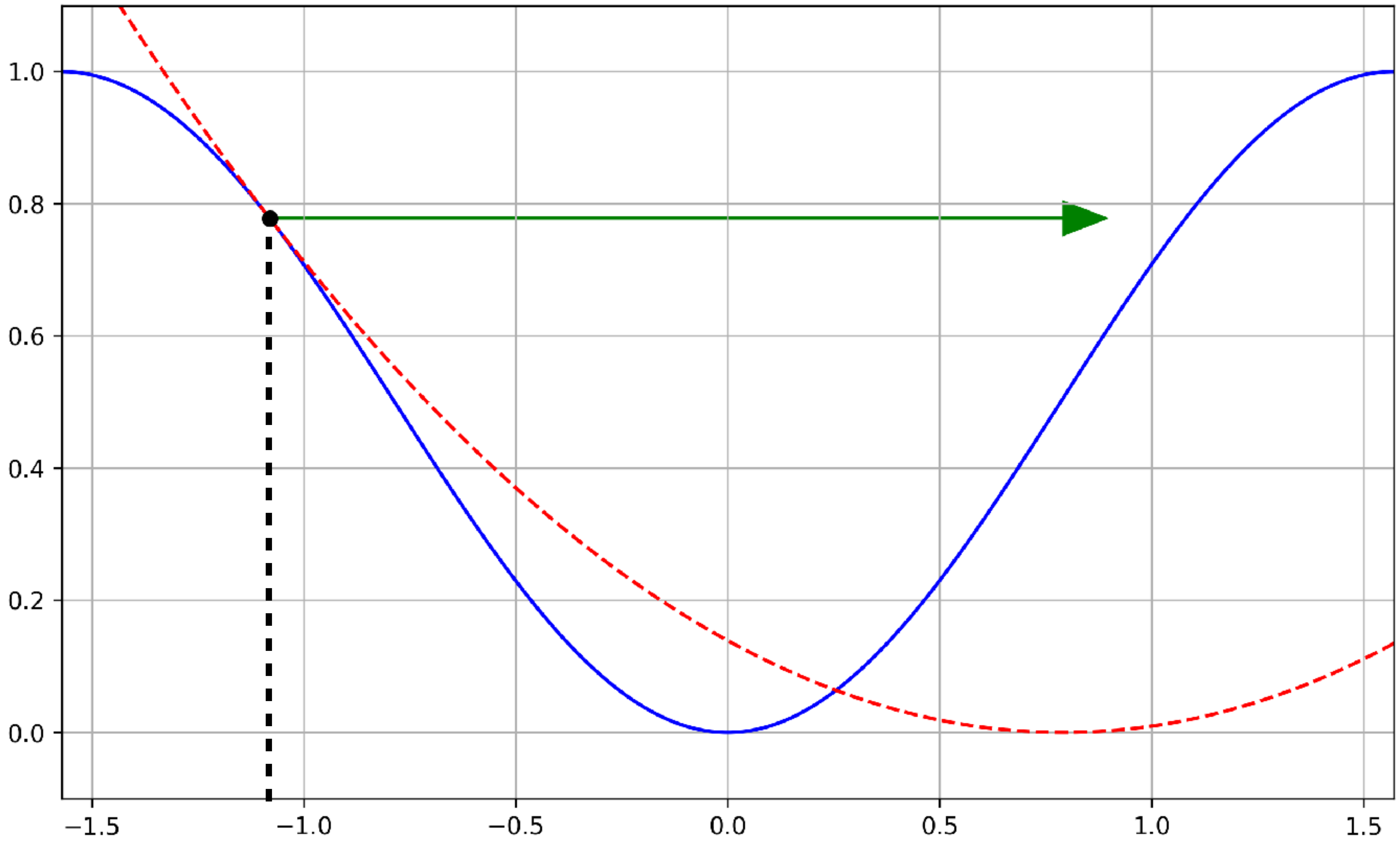
Prerequisites: Gauss-Newton method

$$\Delta x^* = \arg\min_{\Delta x} \left[\sin(1+\Delta x)\right]^2 \approx \arg\min_{\Delta x} \left[\sin(1) + \cos(1)\Delta x\right]^2 = -\frac{\sin(1)}{\cos(1)} = -\tan(1)$$



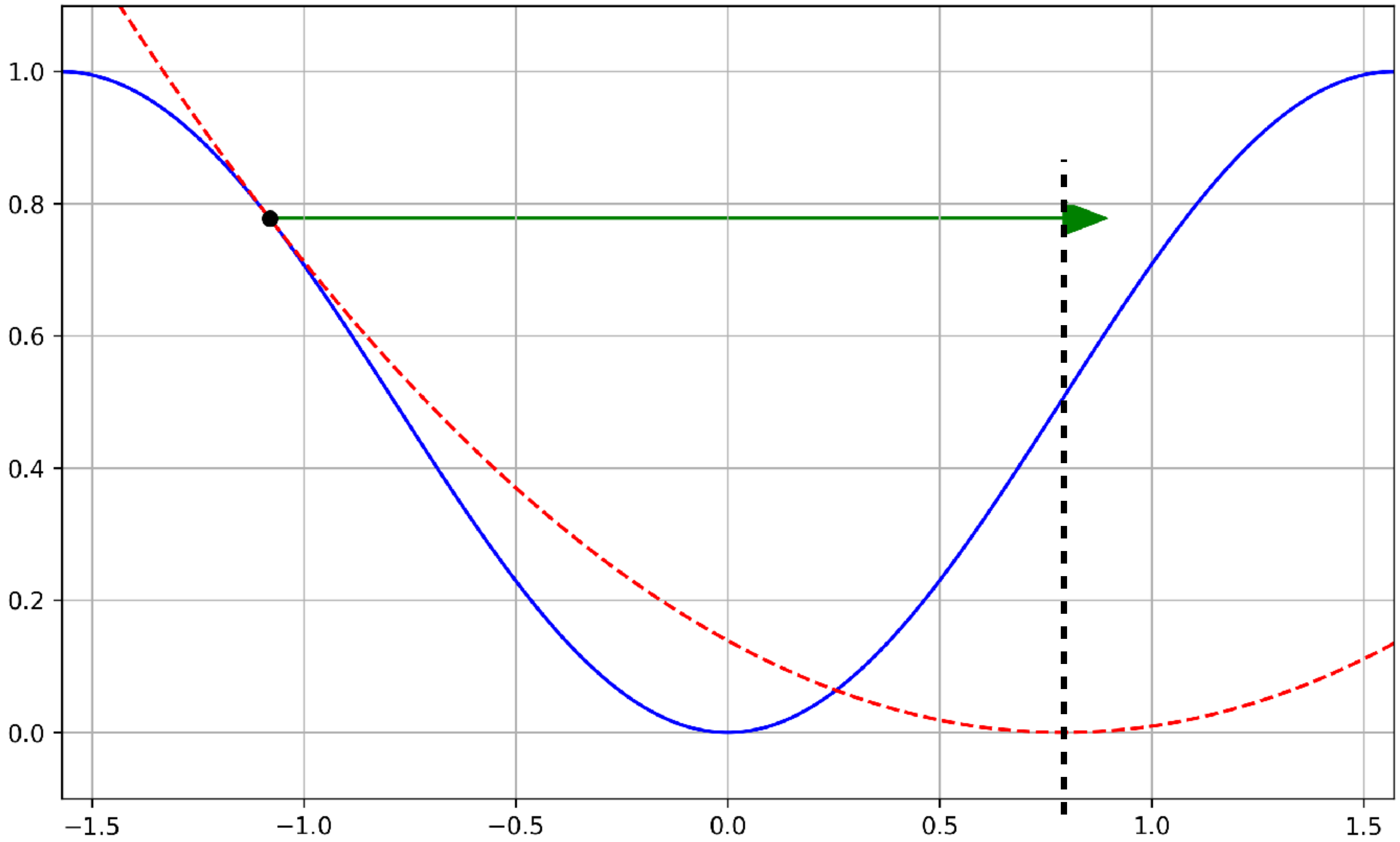
Prerequisites: Gauss-Newton method

$$\Delta x^* = \arg\min_{\Delta x} \left[\sin(1+\Delta x)\right]^2 \approx \arg\min_{\Delta x} \left[\sin(1) + \cos(1)\Delta x\right]^2 = -\frac{\sin(1)}{\cos(1)} = -\tan(1)$$



Prerequisites: Gauss-Newton method

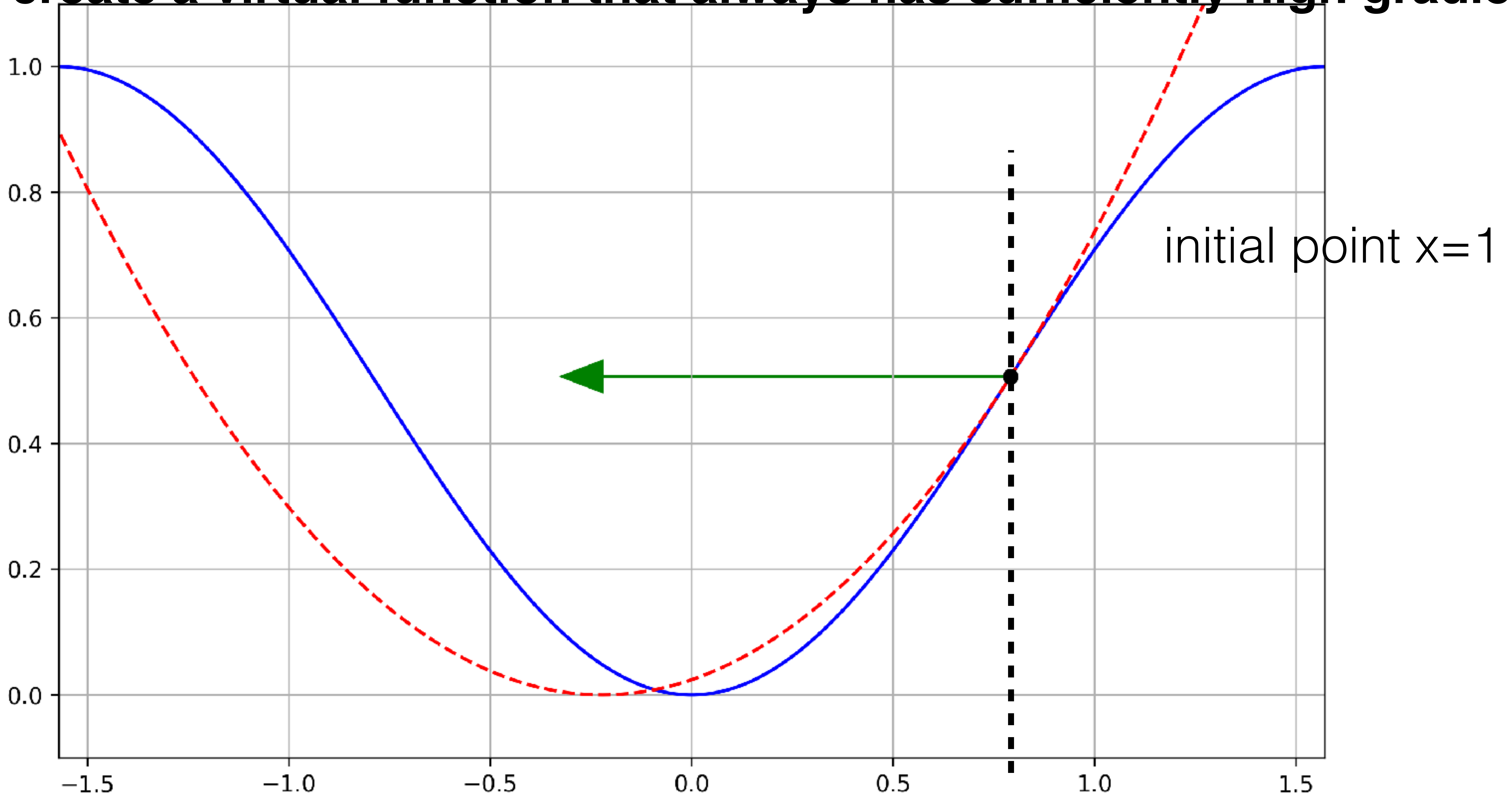
$$\Delta x^* = \arg\min_{\Delta x} \left[\sin(1+\Delta x)\right]^2 \approx \arg\min_{\Delta x} \left[\sin(1)+\cos(1)\Delta x\right]^2 = -\frac{\sin(1)}{\cos(1)} = -\tan(1)$$



Prerequisites: Gauss-Newton method

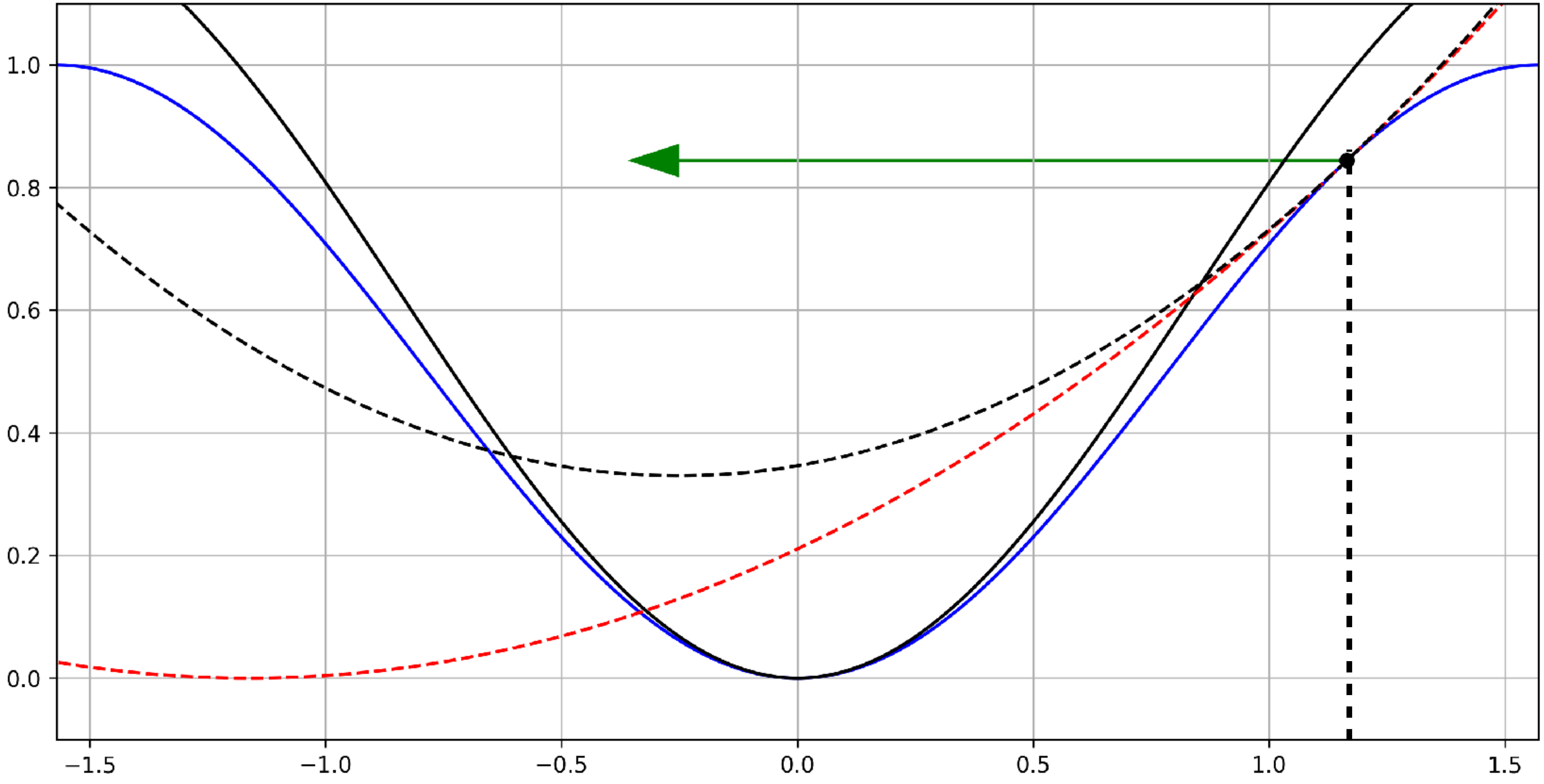
$$\Delta x^* = \arg\min_{\Delta x} \left[\sin(1+\Delta x)\right]^2 \approx \arg\min_{\Delta x} \left[\sin(1) + \cos(1)\Delta x\right]^2 = -\frac{\sin(1)}{\cos(1)} = -\tan(1)$$

Let's create a virtual function that always has sufficiently high gradients



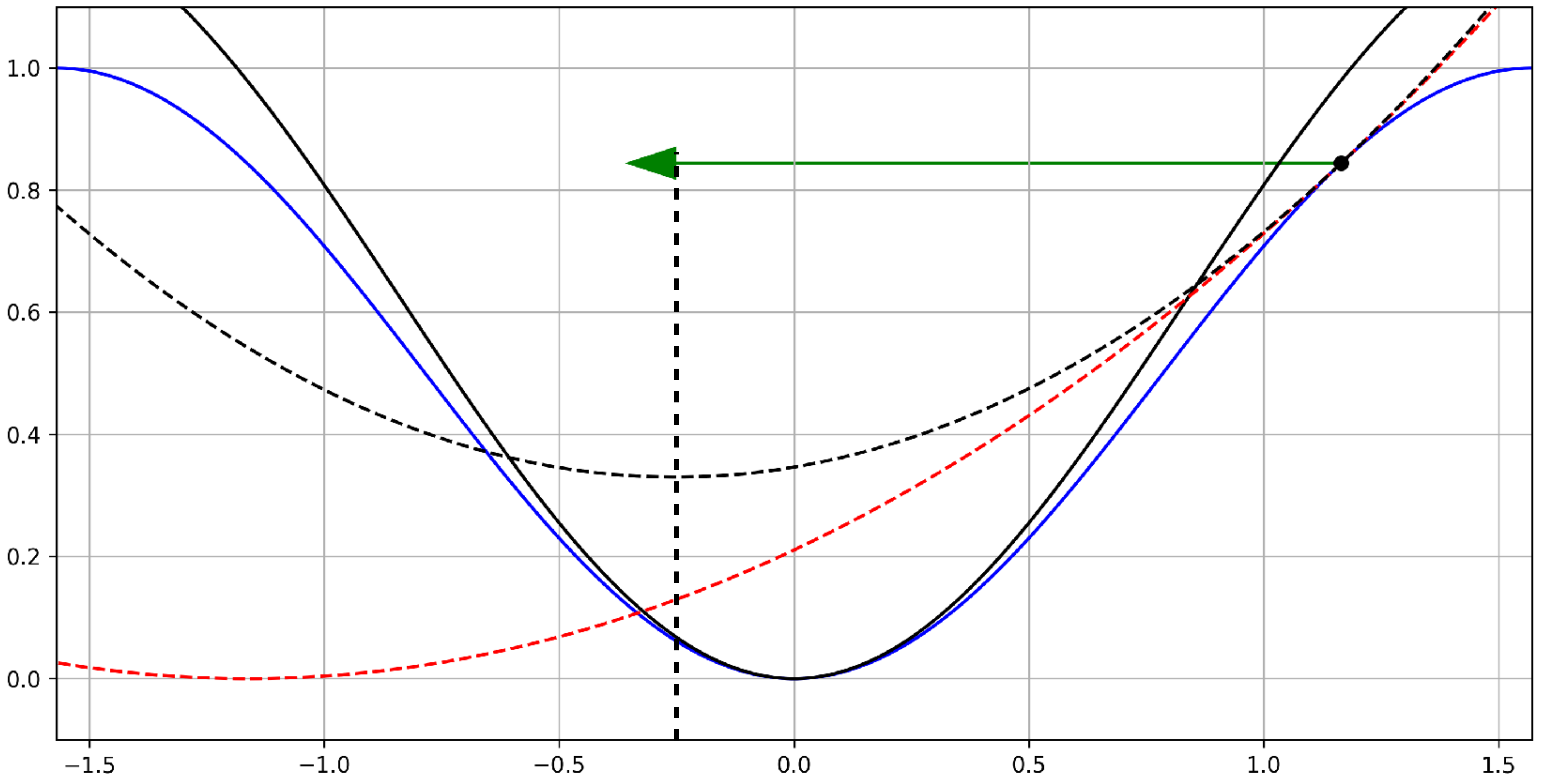
Prerequisites: Levenberg-Marquardt method

$$\Delta x^* = \arg\min_{\Delta x} \left[\sin(x+\Delta x)\right]^2 + \lambda \Delta x^2 \;\approx \arg\min_{\Delta x} \left[\sin(x) + \cos(x)\Delta x\right]^2 + \lambda \Delta x^2 = -\frac{\cos(x)\sin(x)}{\cos(x)^2 + \lambda}$$



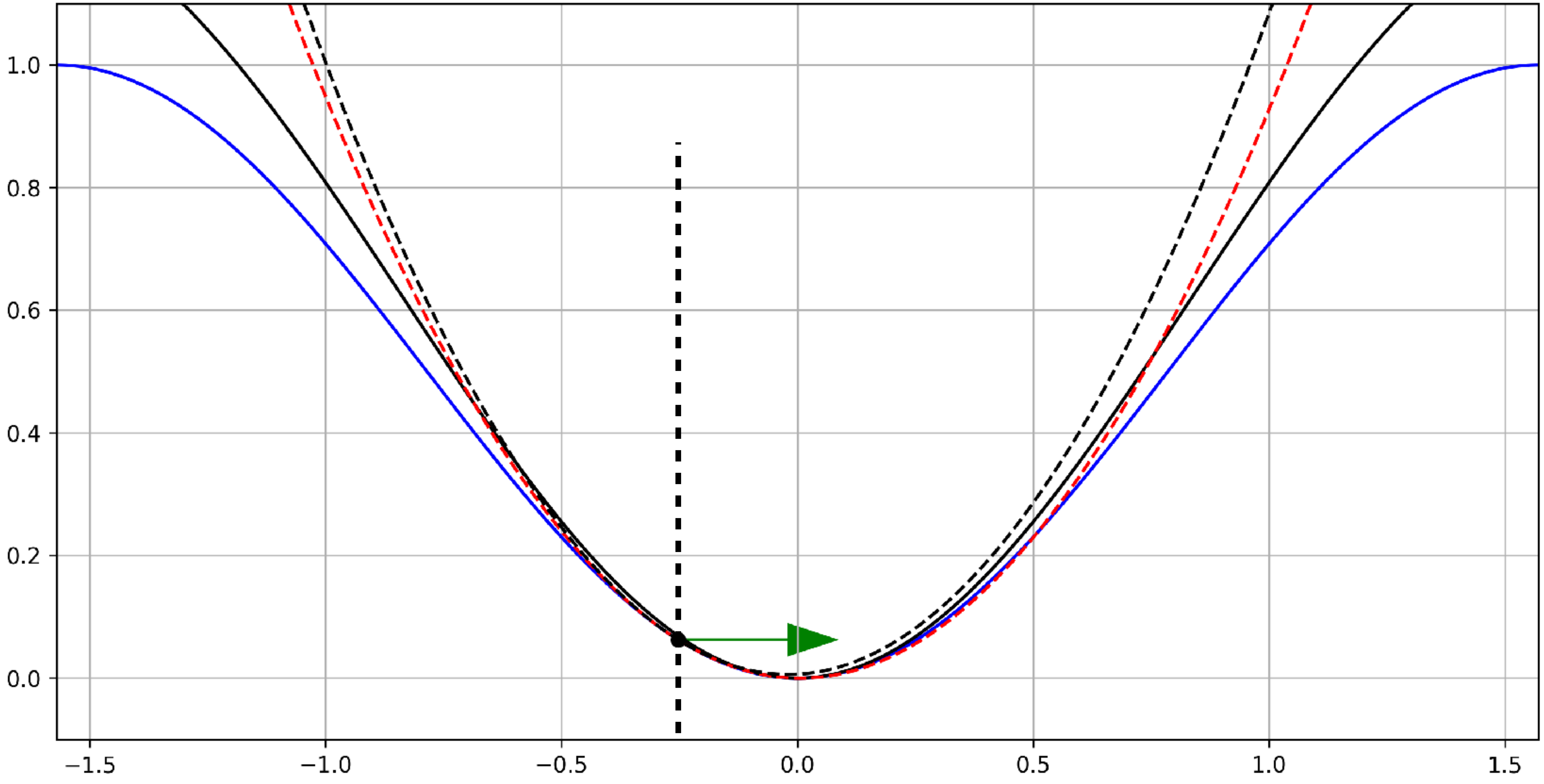
Prerequisites: Levenberg-Marquardt method

$$\Delta x^* = \arg\min_{\Delta x} \left[\sin(x+\Delta x)\right]^2 + \lambda \Delta x^2 \approx \arg\min_{\Delta x} \left[\sin(x) + \cos(x)\Delta x\right]^2 + \lambda \Delta x^2 = -\frac{\cos(x)\sin(x)}{\cos(x)^2 + \lambda}$$



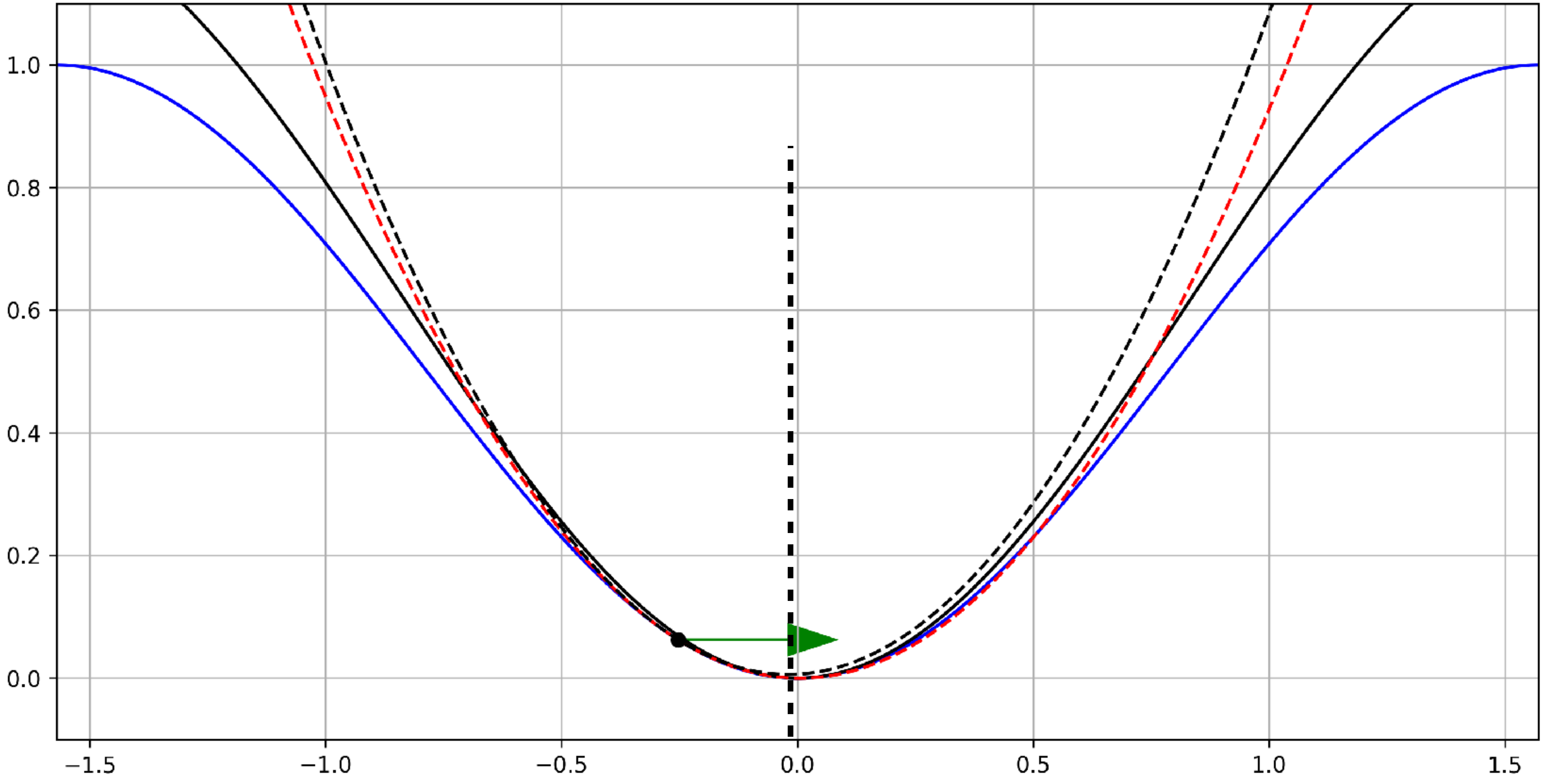
Prerequisites: Levenberg-Marquardt method

$$\Delta x^* = \arg\min_{\Delta x} \left[\sin(x+\Delta x)\right]^2 + \lambda \Delta x^2 \;\approx \arg\min_{\Delta x} \left[\sin(x) + \cos(x)\Delta x\right]^2 + \lambda \Delta x^2 = -\frac{\cos(x)\sin(x)}{\cos(x)^2 + \lambda}$$



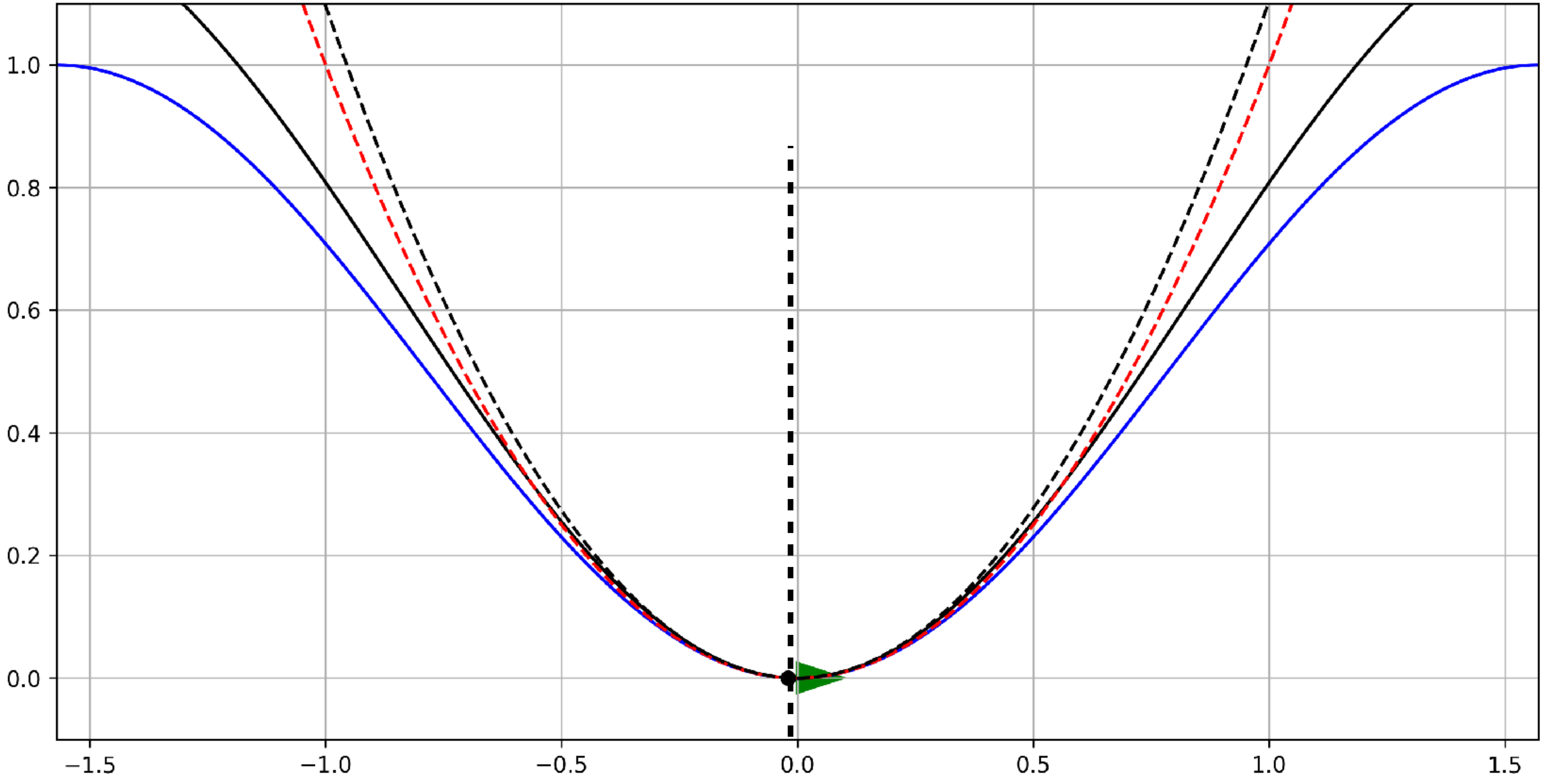
Prerequisites: Levenberg-Marquardt method

$$\Delta x^* = \arg\min_{\Delta x} \left[\sin(x + \Delta x)\right]^2 + \lambda \Delta x^2 \;\approx \arg\min_{\Delta x} \left[\sin(x) + \cos(x)\Delta x\right]^2 + \lambda \Delta x^2 = -\frac{\cos(x)\sin(x)}{\cos(x)^2 + \lambda}$$



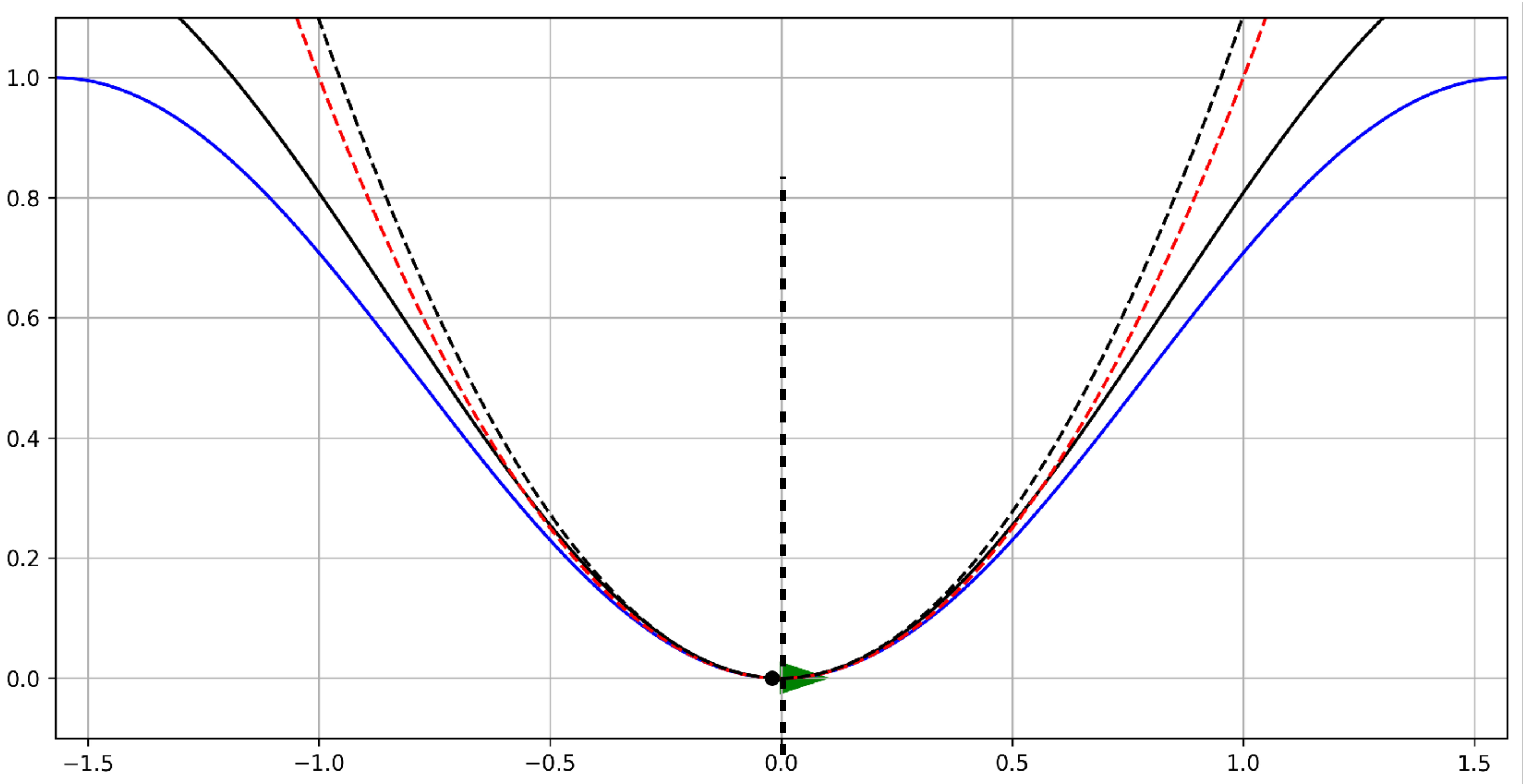
Prerequisites: Levenberg-Marquardt method

$$\Delta x^* = \arg\min_{\Delta x} \left[\sin(x+\Delta x)\right]^2 + \boxed{\lambda \Delta x^2} \approx \arg\min_{\Delta x} \left[\sin(x) + \cos(x)\Delta x\right]^2 + \boxed{\lambda \Delta x^2} = -\frac{\cos(x)\sin(x)}{\cos(x)^2 + \lambda}$$



Prerequisites: Levenberg-Marquardt method

$$\Delta x^* = \arg\min_{\Delta x} \left[\sin(x+\Delta x)\right]^2 + \boxed{\lambda \Delta x^2} \approx \arg\min_{\Delta x} \left[\sin(x) + \cos(x)\Delta x\right]^2 + \boxed{\lambda \Delta x^2} = -\frac{\cos(x)\sin(x)}{\cos(x)^2 + \lambda}$$



Prerequisites: Gauss-Newton method

$$\Delta x^* = \arg\min_{\Delta x} \left[\sin(x + \Delta x)\right]^2 \approx \arg\min_{\Delta x} \left[\sin(x) + \cos(x)\Delta x\right]^2 = -\frac{\sin(x)}{\cos(x)} = -\tan(x)$$

$$\Delta x^* = \arg\min_{\Delta x} \left[f(x + \Delta x)\right]^2 \quad \approx \arg\min_{\Delta x} \left[f(x) + f'(x)\Delta x\right]^2$$

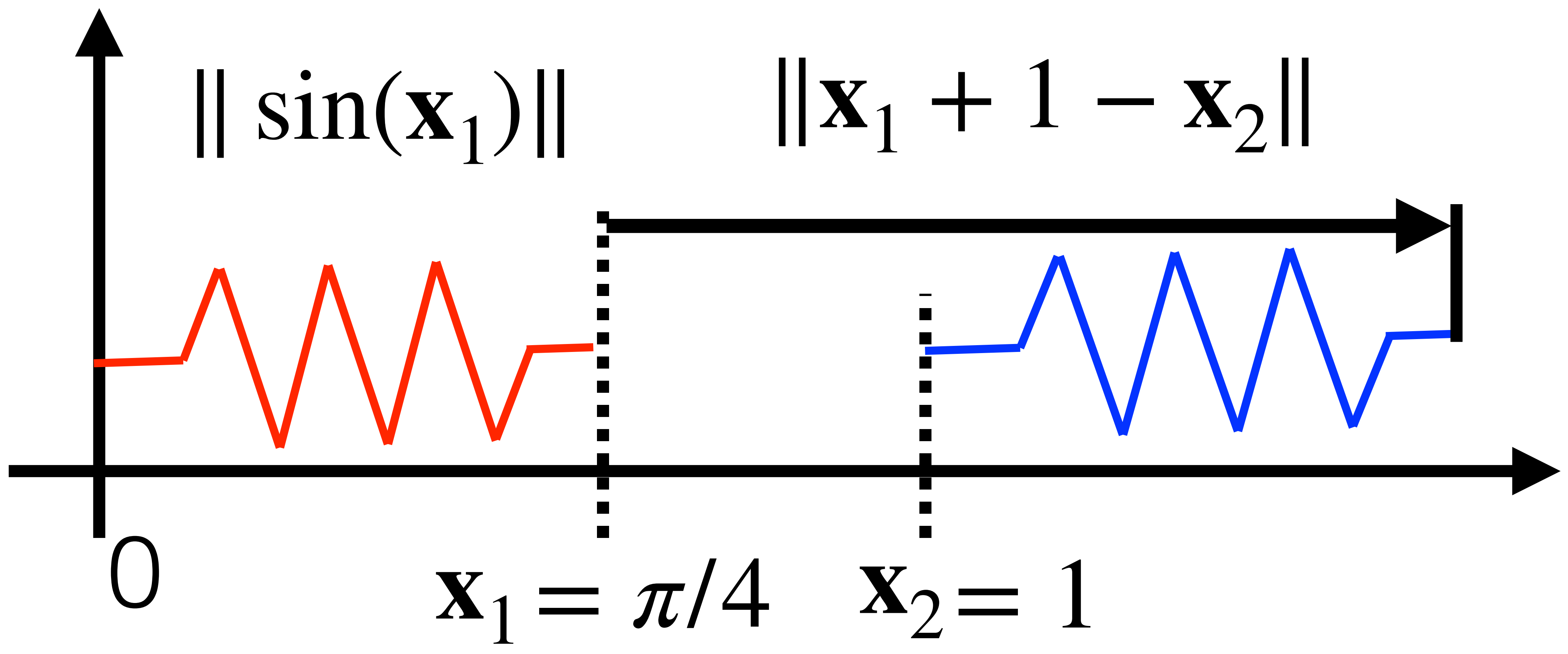
$$= \arg\min_{\Delta x} \underbrace{f'(x)^2}_{\text{Hessian}}\Delta x^2 + 2f'(x)\Delta x + f(x)^2 \quad = \boxed{-\frac{f(x)}{f'(x)}}$$

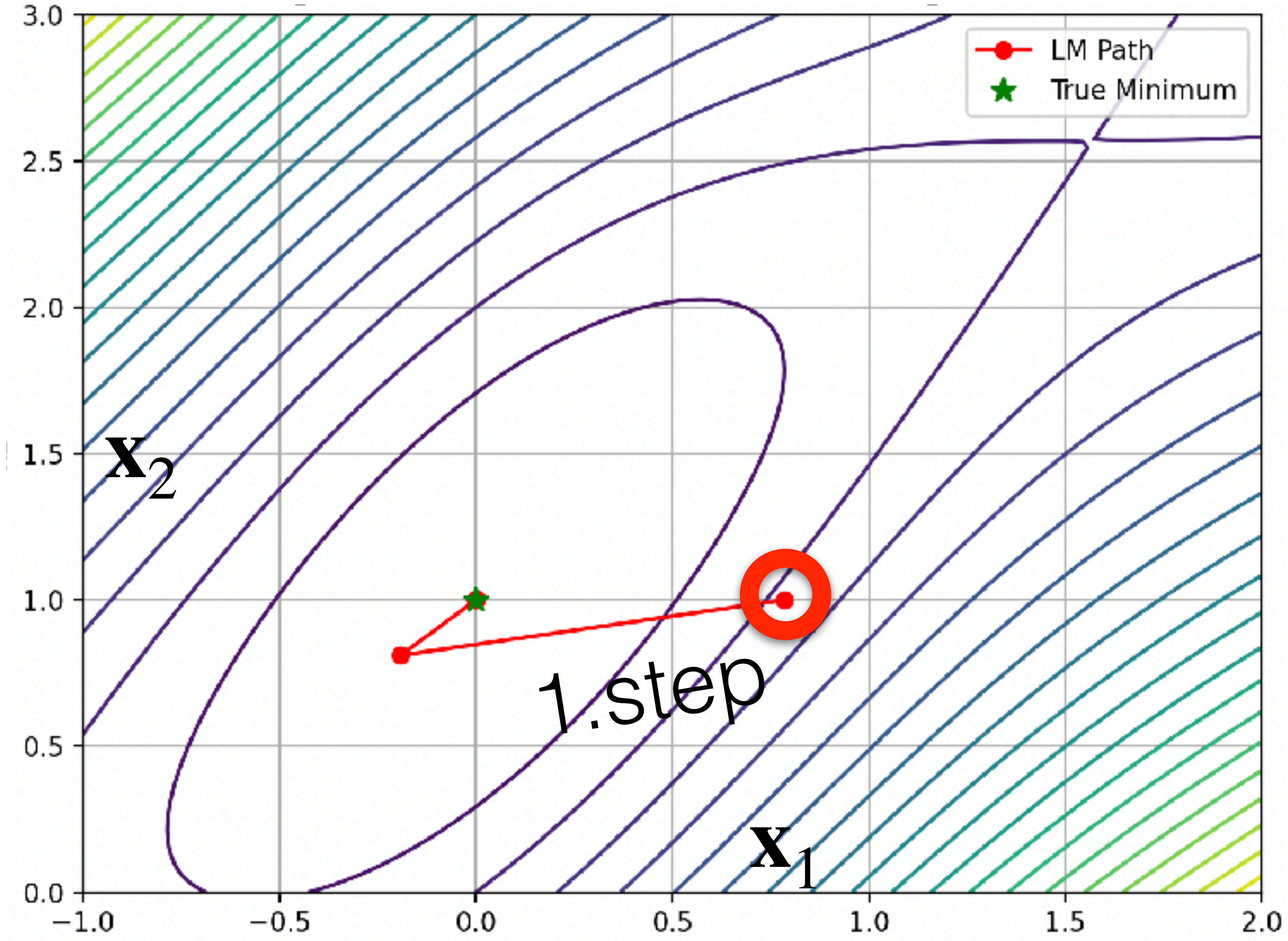
Prerequisites: Levenberg-Marquardt method

$$\Delta x^* = \arg\min_{\Delta x} \left[f(x + \Delta x)\right]^2 \quad \approx \arg\min_{\Delta x} \left[f(x) + f'(x)\Delta x\right]^2 + \lambda \Delta x^2$$

$$= \arg\min_{\Delta x} \underbrace{(f'(x) + \lambda)^2}_{\text{Hessian}}\Delta x^2 + 2f'(x)\Delta x + f(x)^2 = \boxed{-\frac{f(x)f'(x)}{f'(x)^2 + \lambda}}$$

Let's try it on 2D factorgraph problem





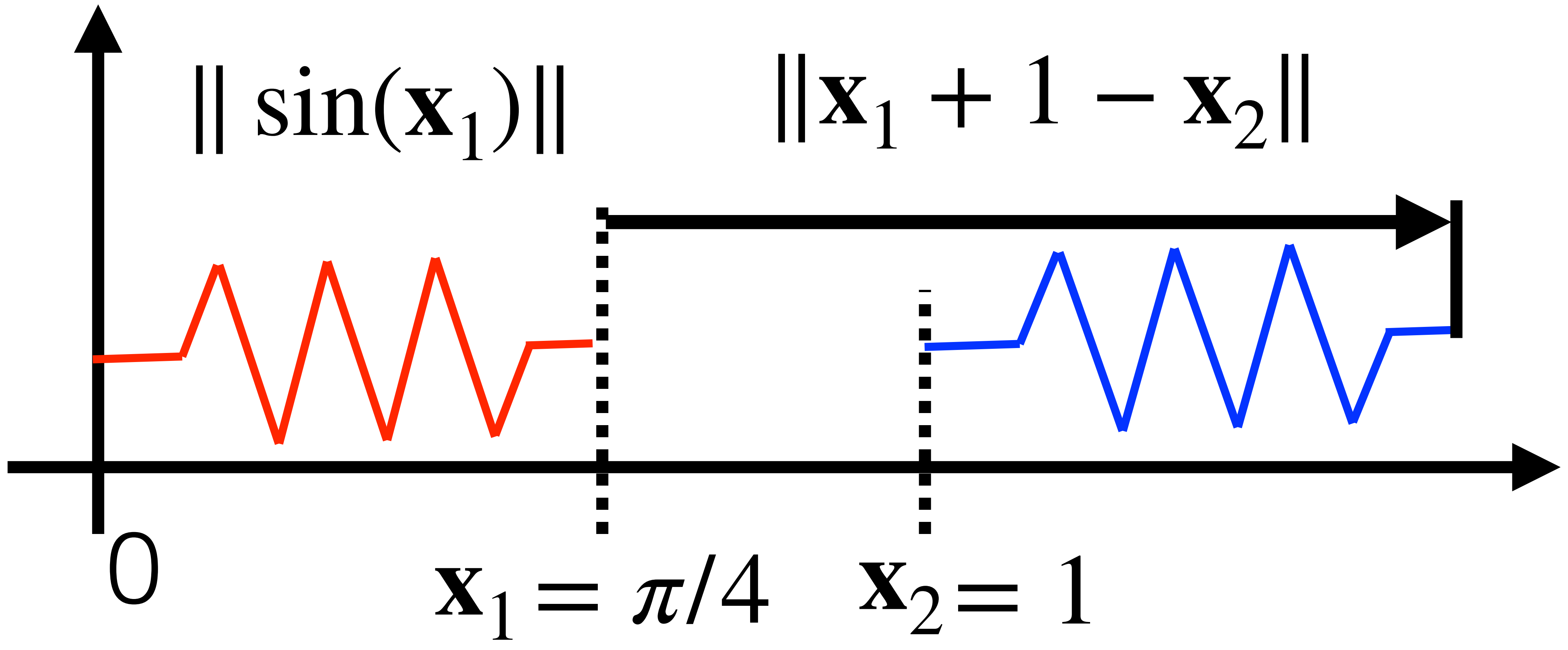
$$\Delta\mathbf{x} = \arg\min_{\mathbf{x}_1,\mathbf{x}_2} \left(\sin\mathbf{x}_1\right)^2 + \left(\mathbf{x}_1 + 1 - \mathbf{x}_2\right)^2$$

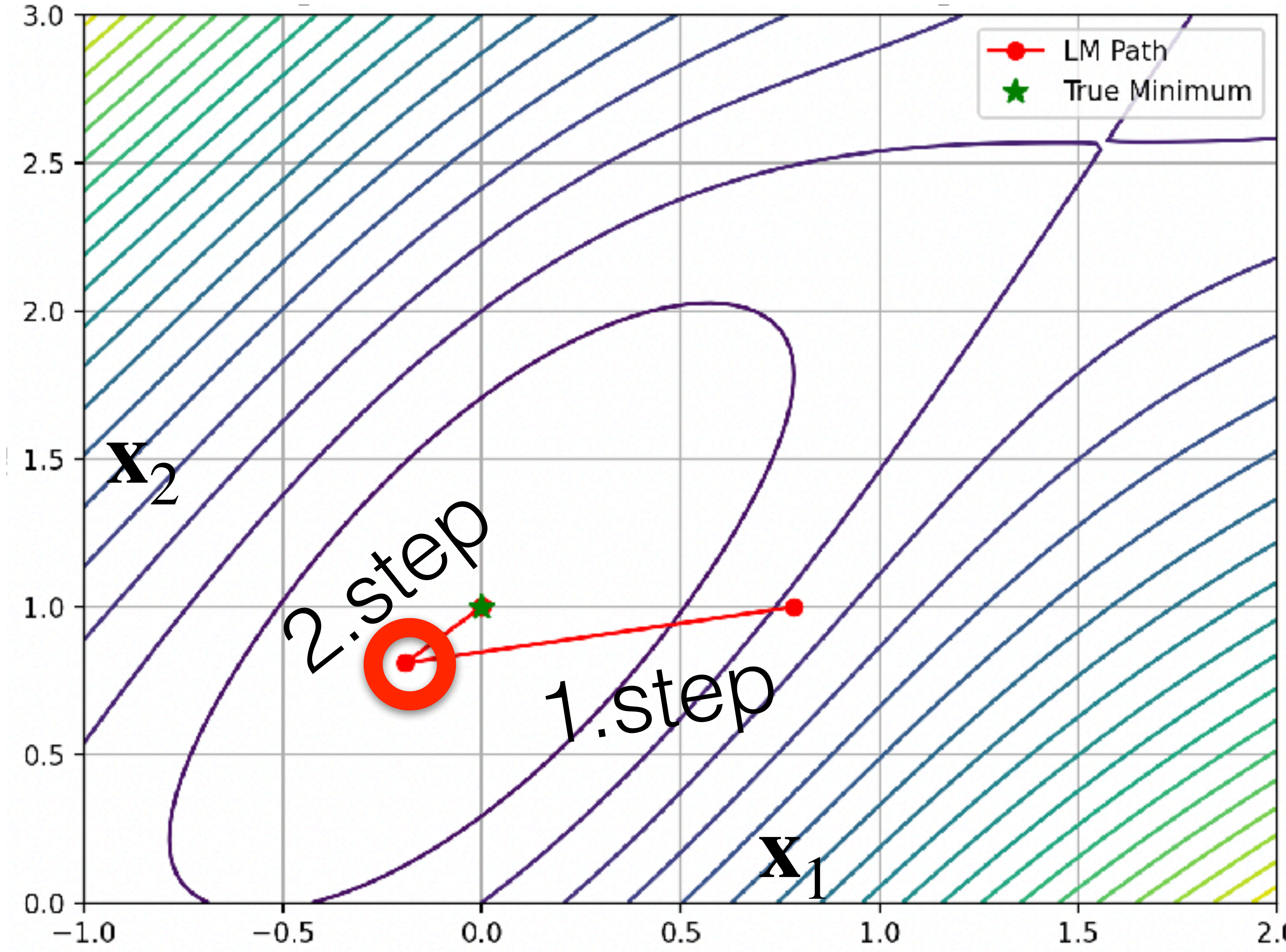
$$= \arg\min_{\mathbf{x}_1,\mathbf{x}_2} \left\| \begin{matrix} \sin\mathbf{x}_1 \\ \mathbf{x}_1 + 1 - \mathbf{x}_2 \end{matrix} \right\|_2^2 = \arg\min_{\Delta\mathbf{x}} \left\| \underbrace{\begin{bmatrix} \sin(\pi/4) \\ \pi/4 + 1 - 1 \end{bmatrix}}_{\mathbf{f}\dots\text{ residuals}} + \underbrace{\begin{bmatrix} \cos(\pi/4) & 0 \\ 1 & -1 \end{bmatrix}}_{\mathbf{J}\dots\text{ jacobian}} \cdot \begin{bmatrix} \Delta\mathbf{x}_1 \\ \Delta\mathbf{x}_2 \end{bmatrix} \right\|_2^2$$

$$= \arg\min_{\Delta\mathbf{x}} \Delta\mathbf{x}^\top \mathbf{J}^\top\mathbf{J}\Delta\mathbf{x} + 2\mathbf{f}^\top\mathbf{J}\Delta\mathbf{x} + \mathbf{f}^\top\mathbf{f} = -(\mathbf{J}^\top\mathbf{J})^{+}\mathbf{J}^\top\mathbf{f} = \begin{bmatrix} 1.5 & -1 \\ -1 & 0 \end{bmatrix}^{-1} \cdot \begin{bmatrix} -1.28 \\ 0.78 \end{bmatrix} = \begin{bmatrix} -1 \\ -0.2 \end{bmatrix}$$

$$2\mathbf{J}^\top\mathbf{J}\Delta\mathbf{x} + 2\mathbf{J}^\top\mathbf{f} = 0 \;\Rightarrow\; \mathbf{J}^\top\mathbf{J}\Delta\mathbf{x} = -\mathbf{J}^\top\mathbf{f} \;\Rightarrow\; \Delta\mathbf{x} = -(\mathbf{J}^\top\mathbf{J})^{+}\mathbf{J}^\top\mathbf{f}$$

Let's try it on 2D factorgraph problem





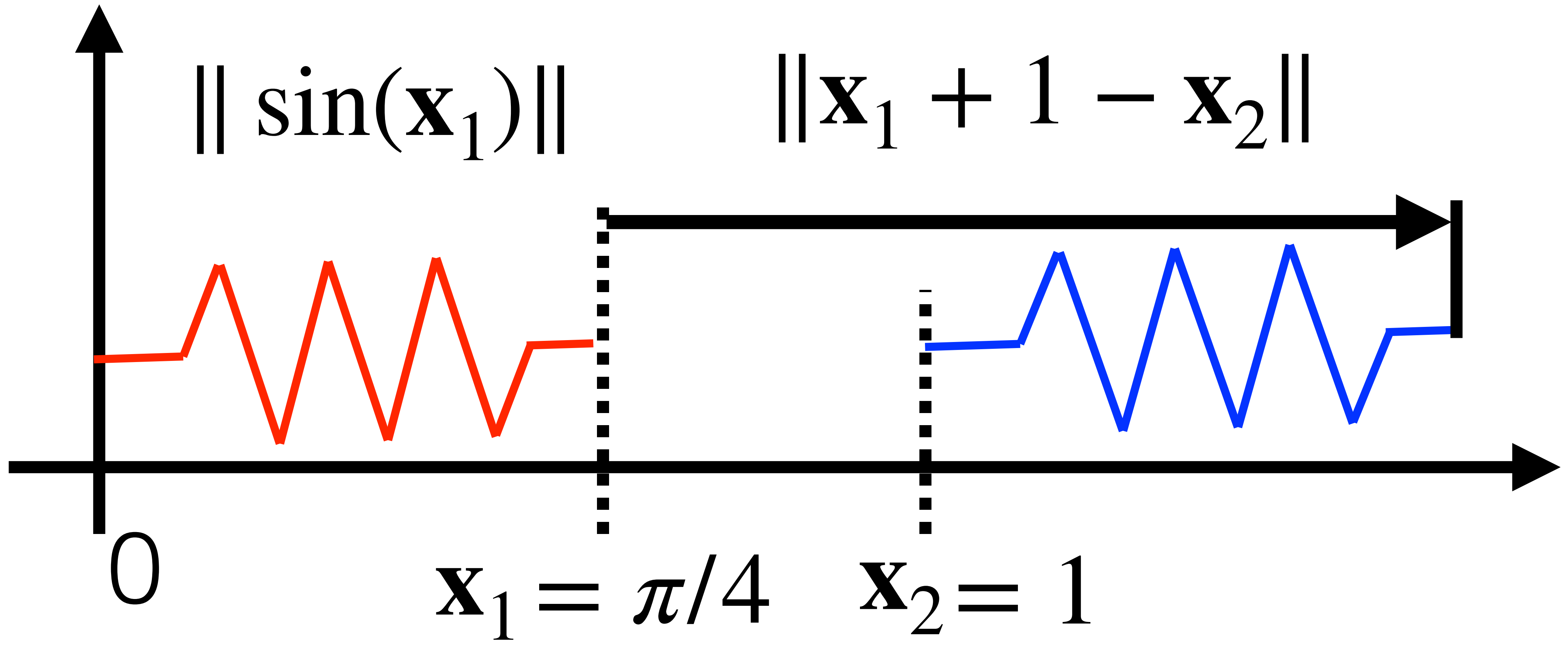
$$\Delta\mathbf{x} = \arg\min_{\mathbf{x}_1,\mathbf{x}_2} \left(\sin \mathbf{x}_1\right)^2 + \left(\mathbf{x}_1 + 1 - \mathbf{x}_2\right)^2$$

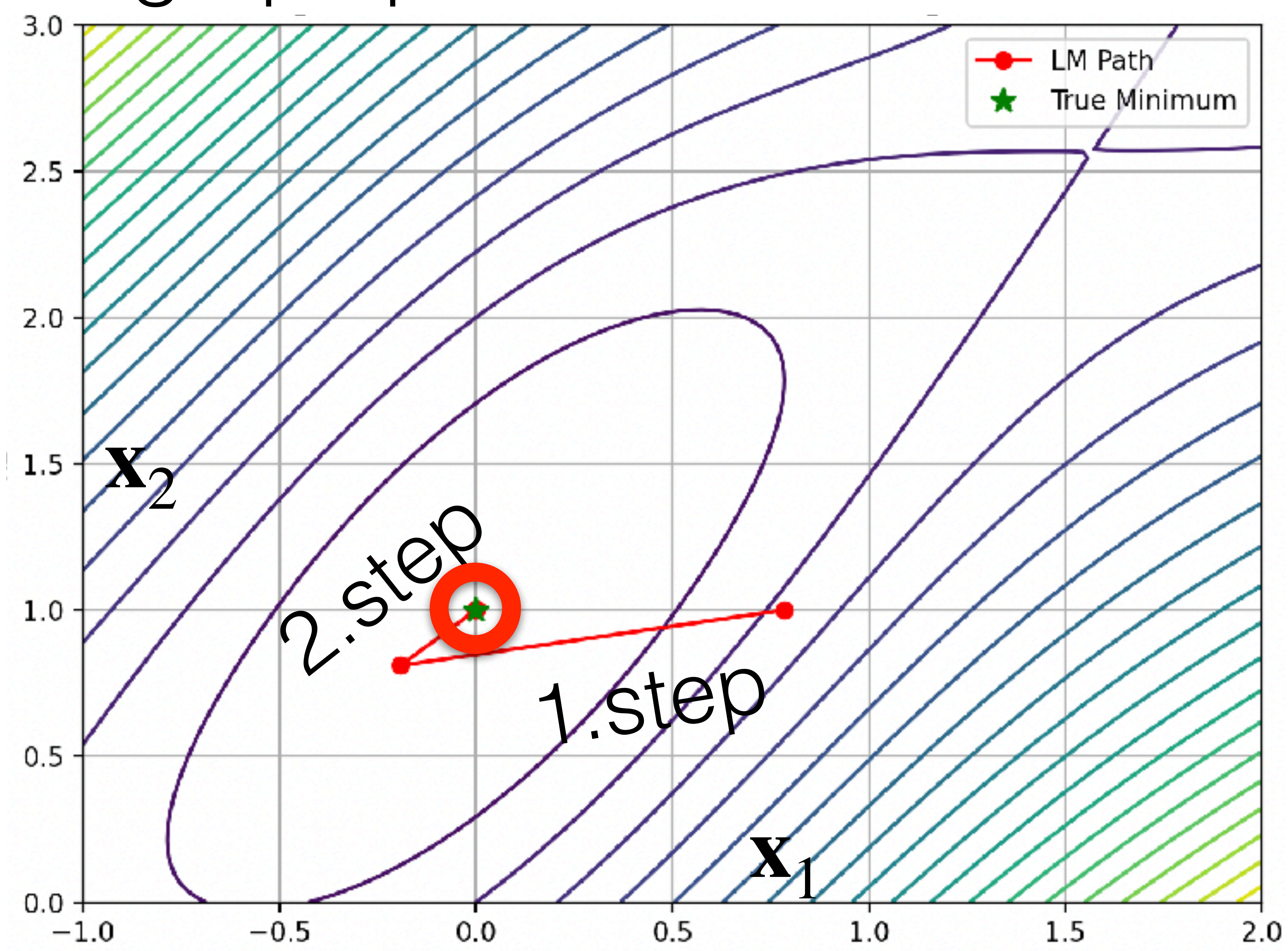
$$= \arg\min_{\mathbf{x}_1,\mathbf{x}_2} \left\| \begin{matrix} \sin \mathbf{x}_1 \\ \mathbf{x}_1 + 1 - \mathbf{x}_2 \end{matrix} \right\|_2^2 = \arg\min_{\Delta\mathbf{x}} \left\| \underbrace{\begin{bmatrix} \sin(\pi/4) \\ \pi/4 + 1 - 1 \end{bmatrix}}_{\mathbf{f}\ldots\ \text{residuals}} + \underbrace{\begin{bmatrix} \cos(\pi/4) & 0 \\ 1 & -1 \end{bmatrix}}_{\mathbf{J}\ldots\ \text{jacobian}} \cdot \begin{bmatrix} \Delta\mathbf{x}_1 \\ \Delta\mathbf{x}_2 \end{bmatrix} \right\|_2^2$$

$$= \arg\min_{\Delta\mathbf{x}} \Delta\mathbf{x}^\top \mathbf{J}^\top \mathbf{J} \Delta\mathbf{x} + 2\mathbf{f}^\top \mathbf{J} \Delta\mathbf{x} + \mathbf{f}^\top \mathbf{f} = -(\mathbf{J}^\top \mathbf{J})^+ \mathbf{J}^\top \mathbf{f} = \begin{bmatrix} 1.5 & -1 \\ -1 & 0 \end{bmatrix}^{-1} \cdot \begin{bmatrix} -1.28 \\ 0.78 \end{bmatrix} = \begin{bmatrix} -1 \\ -0.2 \end{bmatrix}$$

$$2\mathbf{J}^\top \mathbf{J} \Delta\mathbf{x} + 2\mathbf{J}^\top \mathbf{f} = 0 \;\Rightarrow\; \mathbf{J}^\top \mathbf{J} \Delta\mathbf{x} = -\mathbf{J}^\top \mathbf{f} \;\Rightarrow\; \Delta\mathbf{x} = -(\mathbf{J}^\top \mathbf{J})^+ \mathbf{J}^\top \mathbf{f}$$

Let's try it on 2D factorgraph problem





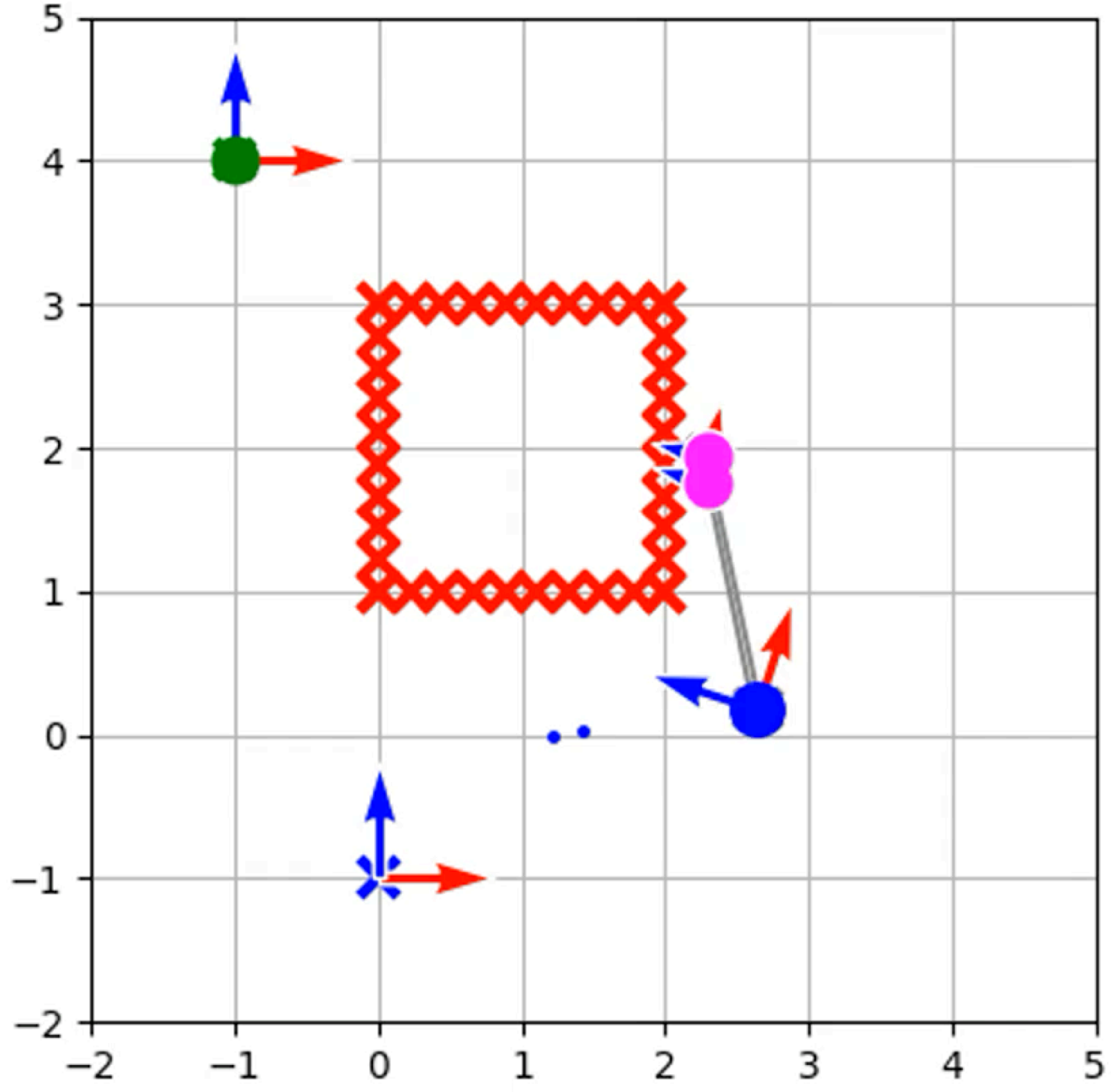
$$\Delta \mathbf{x} = \underset{\mathbf{x}_1,\mathbf{x}_2}{\arg\min} \ \left(\sin \mathbf{x}_1\right)^2 + \left(\mathbf{x}_1 + 1 - \mathbf{x}_2\right)^2$$

$$= \underset{\mathbf{x}_1,\mathbf{x}_2}{\arg\min} \left\| \begin{matrix} \sin \mathbf{x}_1 \\ \mathbf{x}_1 + 1 - \mathbf{x}_2 \end{matrix} \right\|_2^2 = \underset{\Delta \mathbf{x}}{\arg\min} \left\| \underbrace{\begin{bmatrix} \sin(\pi/4) \\ \pi/4 + 1 - 1 \end{bmatrix}}_{\mathbf{f}\text{... residuals}} + \underbrace{\begin{bmatrix} \cos(\pi/4) & 0 \\ 1 & -1 \end{bmatrix}}_{\mathbf{J}\text{... jacobian}} \cdot \begin{bmatrix} \Delta \mathbf{x}_1 \\ \Delta \mathbf{x}_2 \end{bmatrix} \right\|_2^2$$

$$= \underset{\Delta \mathbf{x}}{\arg\min} \ \Delta \mathbf{x}^\top \mathbf{J}^\top \mathbf{J} \Delta \mathbf{x} + 2\mathbf{f}^\top \mathbf{J} \Delta \mathbf{x} + \mathbf{f}^\top \mathbf{f} = -(\mathbf{J}^\top \mathbf{J})^+ \mathbf{J}^\top \mathbf{f} = \begin{bmatrix} 1.5 & -1 \\ -1 & 0 \end{bmatrix}^{-1} \cdot \begin{bmatrix} -1.28 \\ 0.78 \end{bmatrix} = \begin{bmatrix} -1 \\ -0.2 \end{bmatrix}$$

$$2\mathbf{J}^\top \mathbf{J} \Delta \mathbf{x} + 2\mathbf{J}^\top \mathbf{f} = 0 \ \Rightarrow \mathbf{J}^\top \mathbf{J} \Delta \mathbf{x} = -\mathbf{J}^\top \mathbf{f} \ \Rightarrow \Delta \mathbf{x} = -(\mathbf{J}^\top \mathbf{J})^+ \mathbf{J}^\top \mathbf{f}$$

graph-SLAM



$\mathbf{x}_t$ .... estimated robot poses

$\mathbf{m}^{\mathrm{rel}}$....... estimated rel. marker position

$\mathbf{m}^{\mathrm{abs}}$....... known abs. marker position

$\mathbf{z}_t^{\mathbf{m}^{\mathrm{rel}}}, \mathbf{z}_t^{\mathbf{m}^{\mathrm{abs}}}$... marker measurements

...... local coordinate frame

...... ground truth pointcloud map

...... ground truth trajectory

graph-SLAM formulations

$$\mathbf{x}^\star = \arg\min_{\underbrace{\boxed{\substack{\mathbf{x}_0,\dots\mathbf{x}_T \\ \mathbf{m}^1\dots\mathbf{m}^J}}}_{\mathbf{x}}} \overset{\text{GPS}}{\sum_t \|\mathbf{x}_t - \mathbf{z}_t^{gps}\|^2_{\Sigma_t^{gps}}} + \overset{\text{odometry}}{\sum_t \|\mathrm{w2r}(\mathbf{x}_{t+1}, \mathbf{x}_t) - \mathbf{z}^v\|^2_{\Sigma_t^v}} + \overset{\text{marker(s)}}{\sum_{t,j} \|\mathrm{w2r}(\mathbf{m}^j, \mathbf{x}_t) - \mathbf{z}\|^2_{\Sigma}}$$

$$+ \overset{\text{priors}}{\sum_t \|\mathbf{x}_t - \mathbf{x}_t^{prior}\|^2_{\Sigma_t^{prior}}} + \overset{\text{motion model}}{\sum_t \|g(\mathbf{x}_{t-1}, \mathbf{u}_t) - \mathbf{x}_t\|^2_{\Sigma_t^g}} + \overset{\text{loop-closures}}{\sum_t \|\mathrm{w2r}(\mathbf{x}_0, \mathbf{x}_T)\|^2_{\Sigma_t^{lc}}}$$

Optimization

$$\mathbf{x}^\star = \arg \min_{\underbrace{\substack{\mathbf{x}_0,\ldots\mathbf{x}_T \\ \mathbf{m}^1\ldots\mathbf{m}^J}}_{\mathbf{x}}} \sum_t \|\mathbf{x}_t - \mathbf{z}_t^{gps}\|^2_{\Sigma_t^{gps}} + \sum_t \|\mathrm{w2r}(\mathbf{x}_{t+1}, \mathbf{x}_t) - \mathbf{z}^v\|^2_{\Sigma_t^v} + \sum_{t,j} \|\mathrm{w2r}(\mathbf{m}^j, \mathbf{x}_t) - \mathbf{z}\|^2_{\Sigma_t^m}$$

$$+ \sum_t \|\mathbf{x}_t - \mathbf{x}_t^{prior}\|^2_{\Sigma_t^{prior}} + \sum_t \|g(\mathbf{x}_{t-1}, \mathbf{u}_t) - \mathbf{x}_t\|^2_{\Sigma_t^g} + \sum_t \|\mathrm{w2r}(\mathbf{x}_0, \mathbf{x}_T)\|^2_{\Sigma_t^{lc}}$$

n-unknowns, m-residuals=> f-dim?

$$= \arg\min_{\mathbf{x}} \sum_i \|f_i(\mathbf{x})\|^2 = \arg\min_{\mathbf{x}} \left\| \begin{matrix} f_1(\mathbf{x}) \\ \vdots \\ f_N(\mathbf{x}) \end{matrix} \right\|^2 = \arg\min_{\mathbf{x}} \|f(\mathbf{x})\|^2 \quad \text{where} \quad f(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^m$$

vector of residuals $\qquad f'(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^{m\times n}$

Alternative formulation: $\arg\min_{\Delta\mathbf{x}} \|f(\mathbf{x}_k + \Delta\mathbf{x})\|^2$ where $\mathbf{x}_k$ is an initial solution

$$\approx \arg\min_{\Delta\mathbf{x}} \|f(\mathbf{x}_k) + f'(\mathbf{x}_k)\Delta\mathbf{x})\|^2 = -[f'(\mathbf{x}_k)]^+ f(\mathbf{x}_k)$$

GN: $\mathbf{x}_{k+1} = \mathbf{x}_k - [f'(\mathbf{x}_k)]^+ f(\mathbf{x}_k)$

$$\approx \arg\min_{\Delta\mathbf{x}} \|f(\mathbf{x}_k) + f'(\mathbf{x}_k)\Delta\mathbf{x})\|^2 + \lambda\|\Delta\mathbf{x}\|^2 = -[f'(\mathbf{x}_k) + \lambda\mathbf{I}]^+ f(\mathbf{x}_k)$$

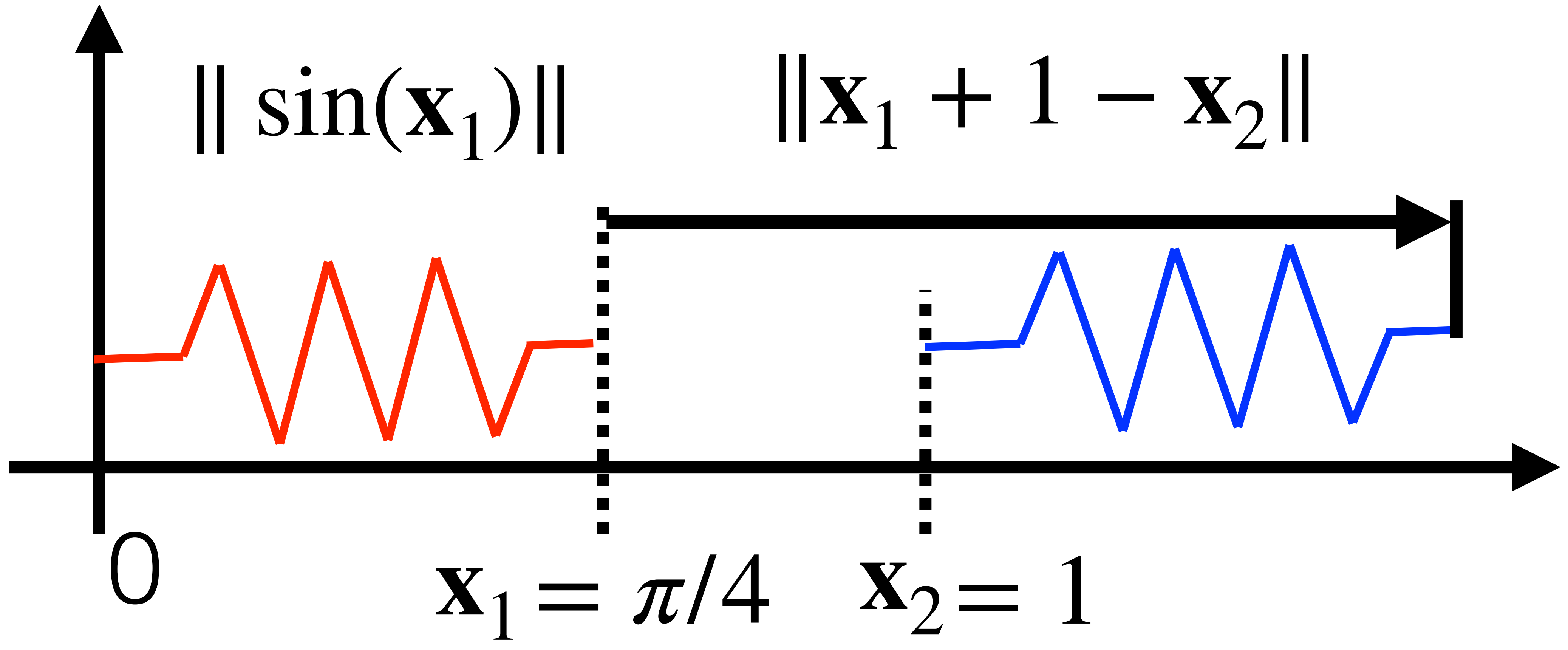
subject to $\|\Delta\mathbf{x}\|^2 \leq c$

various adjustements

LM:
TR: $\mathbf{x}_{k+1} = \mathbf{x}_k - [f'(\mathbf{x}_k) + \lambda\mathbf{I}]^+ f(\mathbf{x}_k)$

```
scipy.optimize.least_squares(fun, x0, jac, method='lm')
```

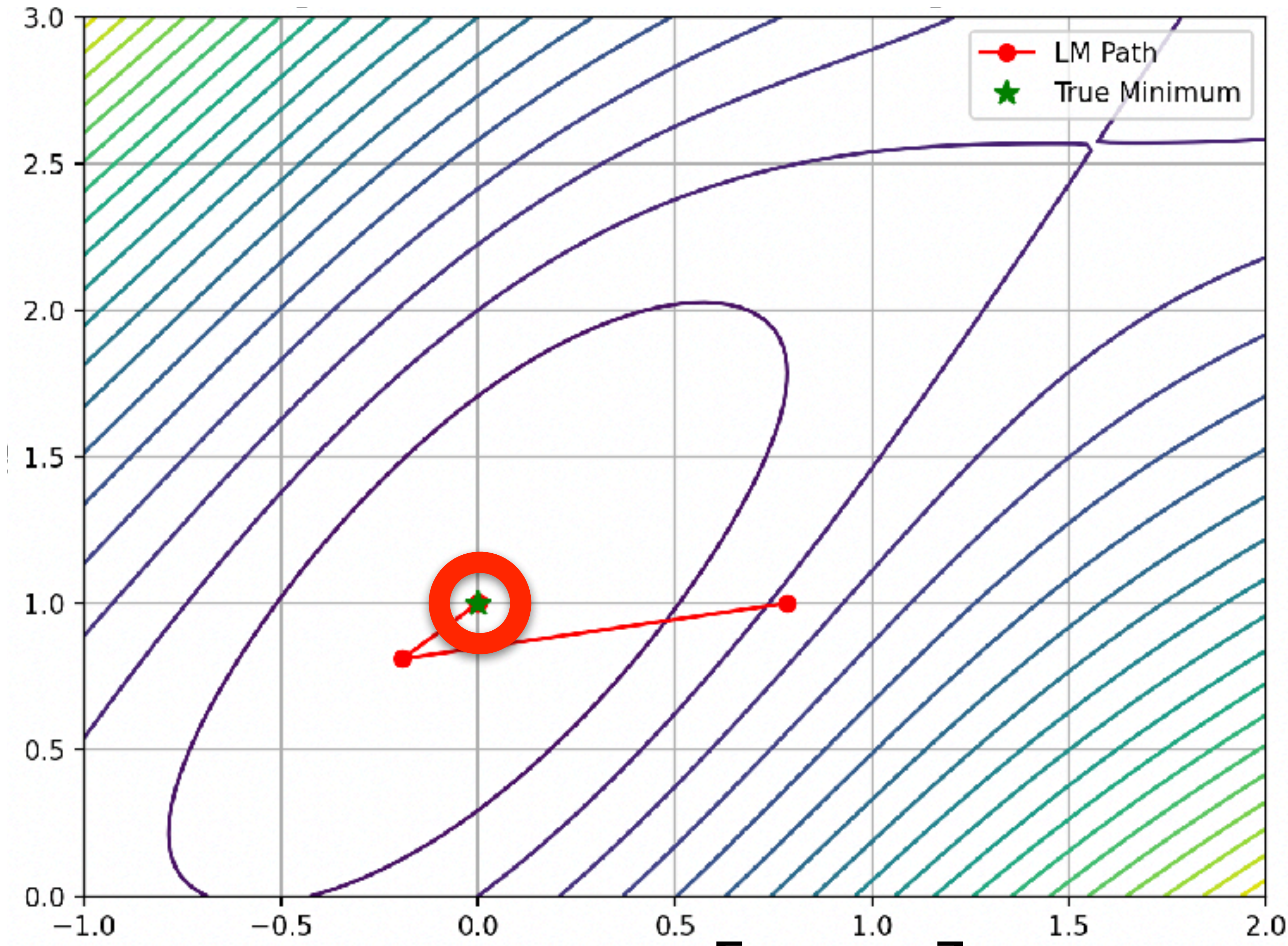
Can we get more efficient solution?



$$\mathbf{f} = \begin{bmatrix} \sin(\pi/4) \\ \pi/4 + 1 - 1 \end{bmatrix}$$

$$\mathbf{J} = \begin{bmatrix} \cos(\pi/4) & 0 \\ -1 & 1 \end{bmatrix}$$

$$\mathbf{H} = \begin{bmatrix} 1.5 & -1 \\ -1 & 1 \end{bmatrix}$$



$$\Delta\mathbf{x} = \underset{\mathbf{x}_1,\mathbf{x}_2}{\arg\min}\ \left(\sin\mathbf{x}_1\right)^2 + \left(\mathbf{x}_1 + 1 - \mathbf{x}_2\right)^2$$

$$2\mathbf{J}^\top\mathbf{J}\Delta\mathbf{x} + 2\mathbf{J}^\top\mathbf{f} = 0 \Rightarrow \mathbf{J}^\top\mathbf{J}\Delta\mathbf{x} = -\mathbf{J}^\top\mathbf{f} \Rightarrow \cancel{\Delta\mathbf{x} = -(\mathbf{J}^\top\mathbf{J})^{+}\mathbf{J}^\top\mathbf{f} = -\mathbf{J}^{+}\mathbf{f}} = \begin{bmatrix} -1 \\ -0.2 \end{bmatrix}$$

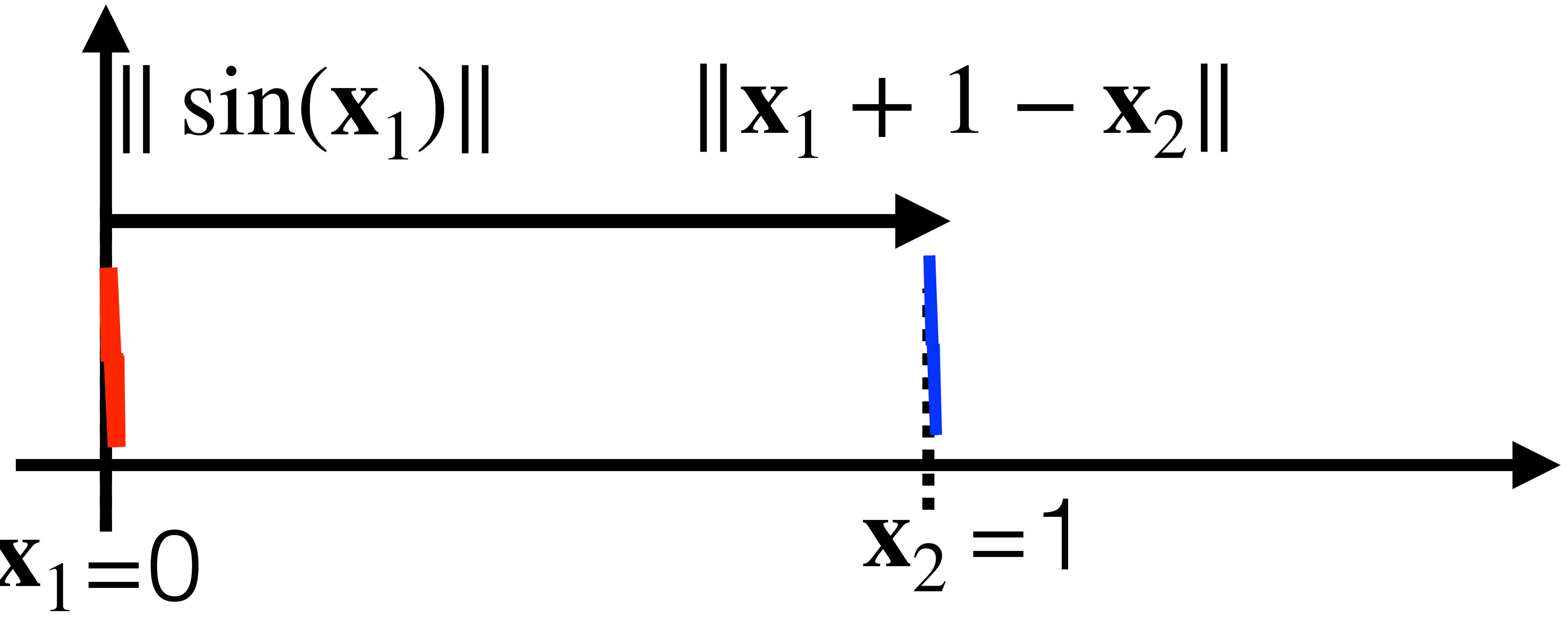
$$\underbrace{\mathbf{J}^\top\mathbf{J}}_{\mathbf{H}}\Delta\mathbf{x} = \underbrace{-\mathbf{J}^\top\mathbf{f}}_{\mathbf{g}} \xRightarrow{\text{Chol } \mathbf{H}} \mathbf{R} = \begin{bmatrix} 1.2 & -0.82 \\ 0 & 0.58 \end{bmatrix} \Rightarrow \mathbf{R}^\top\underbrace{\mathbf{R}\Delta\mathbf{x}}_{\mathbf{y}} = \mathbf{g} \Rightarrow \mathbf{R}^\top\mathbf{y} = \mathbf{g} \Rightarrow \mathbf{y} = \begin{bmatrix} -1.04 \\ -0.12 \end{bmatrix}$$

At convergence point $\mathbf{R} = \begin{bmatrix} 1.4 & -0.7 \\ 0 & 0.7 \end{bmatrix}$

$$\Rightarrow \mathbf{R}\Delta\mathbf{x} = \mathbf{y} \Rightarrow \mathbf{y} = \begin{bmatrix} -1.0 \\ -0.2 \end{bmatrix}$$

It converges to global optimum after 2 iterations

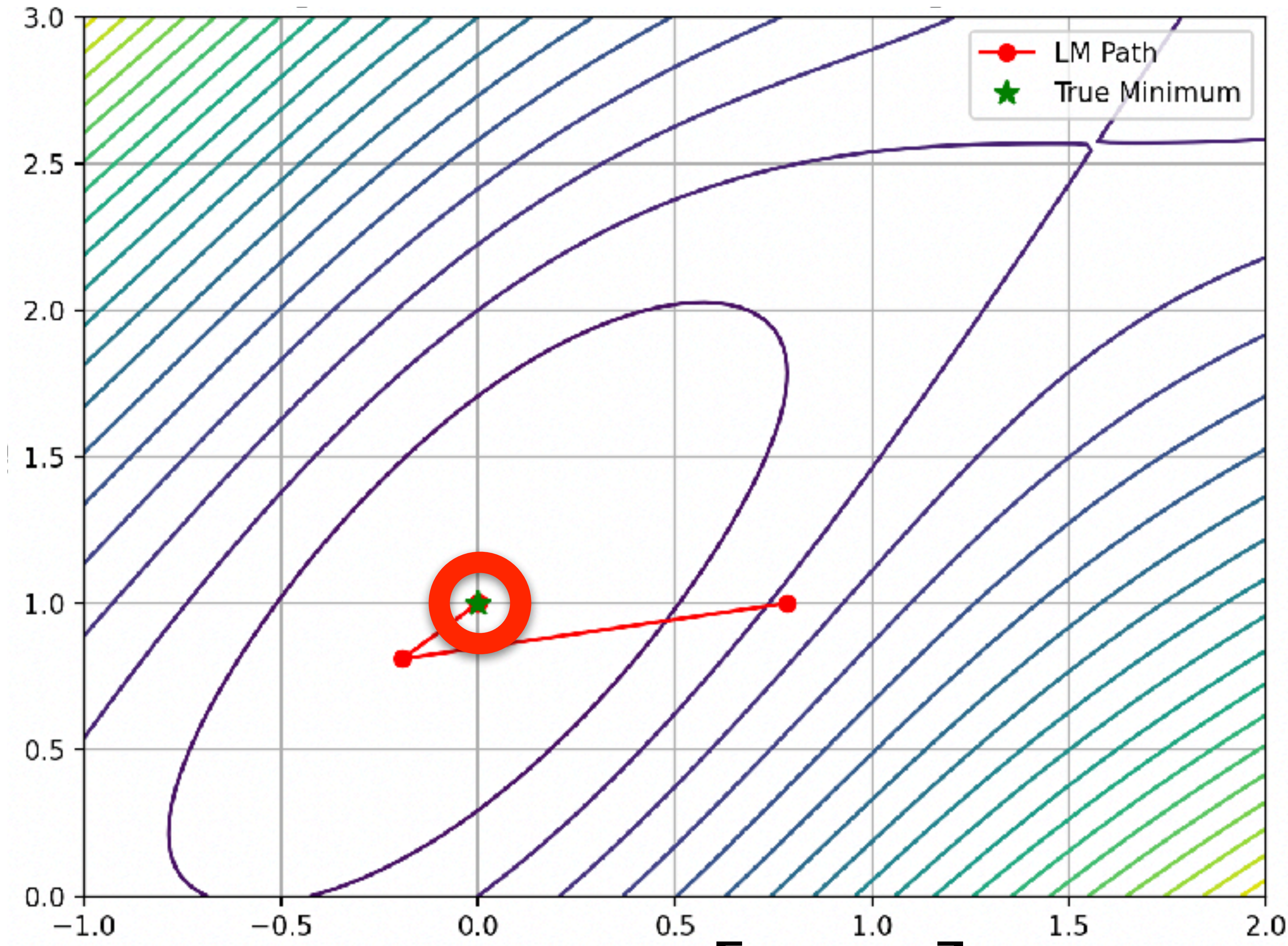
Can we get more efficient solution?



$$\mathbf{f} = \begin{bmatrix} \sin(\pi/4) \\ \pi/4 + 1 - 1 \end{bmatrix}$$

$$\mathbf{J} = \begin{bmatrix} \cos(\pi/4) & 0 \\ -1 & 1 \end{bmatrix}$$

$$\mathbf{H} = \begin{bmatrix} 1.5 & -1 \\ -1 & 1 \end{bmatrix}$$



$$\Delta\mathbf{x} = \arg\min_{\mathbf{x}_1,\mathbf{x}_2} \left(\sin\mathbf{x}_1\right)^2 + \left(\mathbf{x}_1 + 1 - \mathbf{x}_2\right)^2$$

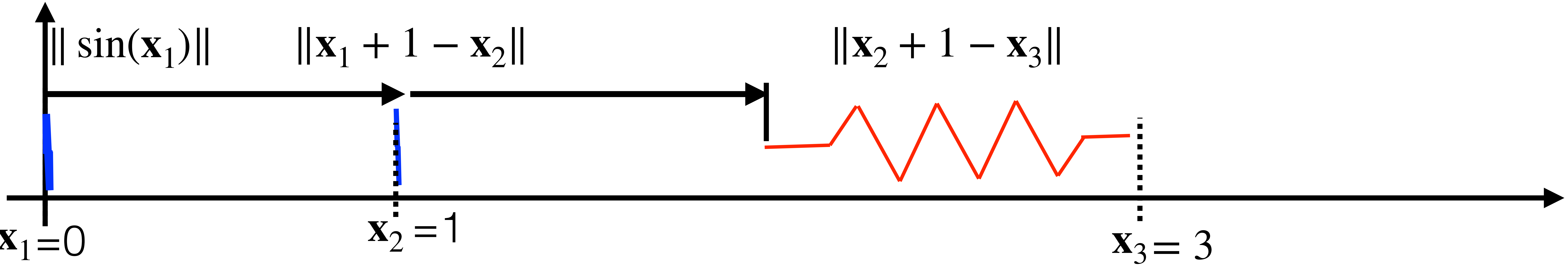
$$2\mathbf{J}^\top\mathbf{J}\Delta\mathbf{x} + 2\mathbf{J}^\top\mathbf{f} = 0 \Rightarrow \mathbf{J}^\top\mathbf{J}\Delta\mathbf{x} = -\mathbf{J}^\top\mathbf{f} \Rightarrow \cancel{\Delta\mathbf{x} = -(\mathbf{J}^\top\mathbf{J})^{+}\mathbf{J}^\top\mathbf{f} = -\mathbf{J}^{+}\mathbf{f}} = \begin{bmatrix} -1 \\ -0.2 \end{bmatrix}$$

$$\underbrace{\mathbf{J}^\top\mathbf{J}}_{\mathbf{H}}\,\Delta\mathbf{x} = \underbrace{-\mathbf{J}^\top\mathbf{f}}_{\mathbf{g}} \xRightarrow{\text{Chol }\mathbf{H}} \mathbf{R} = \begin{bmatrix} 1.2 & -0.82 \\ 0 & 0.58 \end{bmatrix} \Rightarrow \mathbf{R}^\top\underbrace{\mathbf{R}\Delta\mathbf{x}}_{\mathbf{y}} = \mathbf{g} \Rightarrow \mathbf{R}^\top\mathbf{y} = \mathbf{g} \Rightarrow \mathbf{y} = \begin{bmatrix} -1.04 \\ -0.12 \end{bmatrix}$$

At convergence point $\mathbf{R} = \begin{bmatrix} 1.4 & -0.7 \\ 0 & 0.7 \end{bmatrix}$ $\Rightarrow \mathbf{R}\Delta\mathbf{x} = \mathbf{y} \Rightarrow \mathbf{y} = \begin{bmatrix} -1.0 \\ -0.2 \end{bmatrix}$

Let's assume that new measurement has been received

Can we get more efficient solution?



$$\Delta \mathbf{x} = \arg\min_{\mathbf{x}_1,\mathbf{x}_2} \left(\sin \mathbf{x}_1\right)^2 + \left(\mathbf{x}_1 + 1 - \mathbf{x}_2\right)^2$$

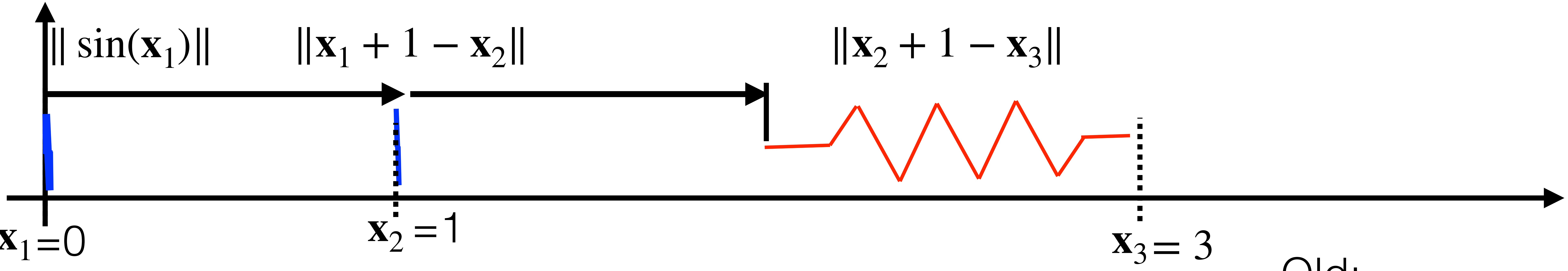
$$2\mathbf{J}^\top\mathbf{J}\Delta\mathbf{x} + 2\mathbf{J}^\top\mathbf{f} = 0 \Rightarrow \mathbf{J}^\top\mathbf{J}\Delta\mathbf{x} = -\mathbf{J}^\top\mathbf{f} \Rightarrow \cancel{\Delta\mathbf{x} = -(\mathbf{J}^\top\mathbf{J})^{+}\mathbf{J}^\top\mathbf{f} = -\mathbf{J}^{+}\mathbf{f}} = \begin{bmatrix} -1 \\ -0.2 \end{bmatrix}$$

$$\underbrace{\mathbf{J}^\top\mathbf{J}}_{\mathbf{H}}\Delta\mathbf{x} = \underbrace{-\mathbf{J}^\top\mathbf{f}}_{\mathbf{g}} \xRightarrow{\text{Chol }\mathbf{H}} \mathbf{R} = \begin{bmatrix} 1.2 & -0.82 \\ 0 & 0.58 \end{bmatrix} \Rightarrow \mathbf{R}^\top\underbrace{\mathbf{R}\Delta\mathbf{x}}_{\mathbf{y}} = \mathbf{g} \Rightarrow \mathbf{R}^\top\mathbf{y} = \mathbf{g} \Rightarrow \mathbf{y} = \begin{bmatrix} -1.04 \\ -0.12 \end{bmatrix}$$

At convergence point $\mathbf{R} = \begin{bmatrix} 1.4 & -0.7 \\ 0 & 0.7 \end{bmatrix}$ $\Rightarrow \mathbf{R}\Delta\mathbf{x} = \mathbf{y} \Rightarrow \mathbf{y} = \begin{bmatrix} -1.0 \\ -0.2 \end{bmatrix}$

Let’s assume that new measurement has been received

Can we get more efficient solution?



$$\Delta\mathbf{x} = \arg\min_{\mathbf{x}_1,\mathbf{x}_2} \left(\sin \mathbf{x}_1\right)^2 + \left(\mathbf{x}_1 + 1 - \mathbf{x}_2\right)^2 + \left(\mathbf{x}_3 + 1 - \mathbf{x}_2\right)^2$$

Old:

$$\mathbf{R} = \begin{bmatrix} 1.4 & -0.7 \\ 0 & 0.7 \end{bmatrix}$$

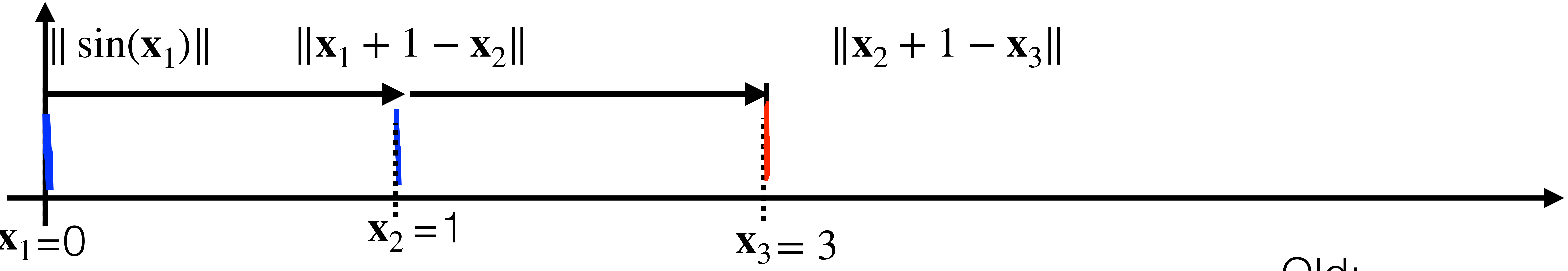
$$\overline{\mathbf{R}} = \begin{bmatrix} 1.4 & -0.7 & 0 \\ 0 & 0.7 & 0 \\ 0 & 1 & -1 \end{bmatrix} \xRightarrow{\text{Givens}} \mathbf{R} = \begin{bmatrix} 1.4 & -0.7 & 0 \\ 0 & 1.2 & -0.8 \\ 0 & 0 & -0.6 \end{bmatrix}$$

$$\Rightarrow \mathbf{R}^\top \mathbf{y} = \mathbf{g} \Rightarrow \mathbf{y} = \begin{bmatrix} 0 \\ 0.8 \\ -1.7 \end{bmatrix}$$

$$\Rightarrow \mathbf{R}\Delta\mathbf{x} = \mathbf{y} \Rightarrow \Delta\mathbf{x} = \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix}$$

I can ignore the old stuff and solve it again but there is more efficient way

Can we get more efficient solution?



Old:

$$\mathbf{R} = \begin{bmatrix} 1.4 & -0.7 \\ 0 & 0.7 \end{bmatrix}$$

$$\Delta\mathbf{x} = \arg\min_{\mathbf{x}_1,\mathbf{x}_2} \left(\sin\mathbf{x}_1\right)^2 + \left(\mathbf{x}_1 + 1 - \mathbf{x}_2\right)^2 + \left(\mathbf{x}_3 + 1 - \mathbf{x}_2\right)^2$$

$$\overline{\mathbf{R}} = \begin{bmatrix} 1.4 & -0.7 & 0 \\ 0 & 0.7 & 0 \\ 0 & 1 & -1 \end{bmatrix} \xRightarrow{\text{Givens}} \mathbf{R} = \begin{bmatrix} 1.4 & -0.7 & 0 \\ 0 & 1.2 & -0.8 \\ 0 & 0 & -0.6 \end{bmatrix}$$

$$\Rightarrow \mathbf{R}^\top\mathbf{y} = \mathbf{g} \Rightarrow \mathbf{y} = \begin{bmatrix} 0 \\ 0.8 \\ -1.7 \end{bmatrix}$$

$$\Rightarrow \mathbf{R}\Delta\mathbf{x} = \mathbf{y} \Rightarrow \Delta\mathbf{x} = \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix}$$

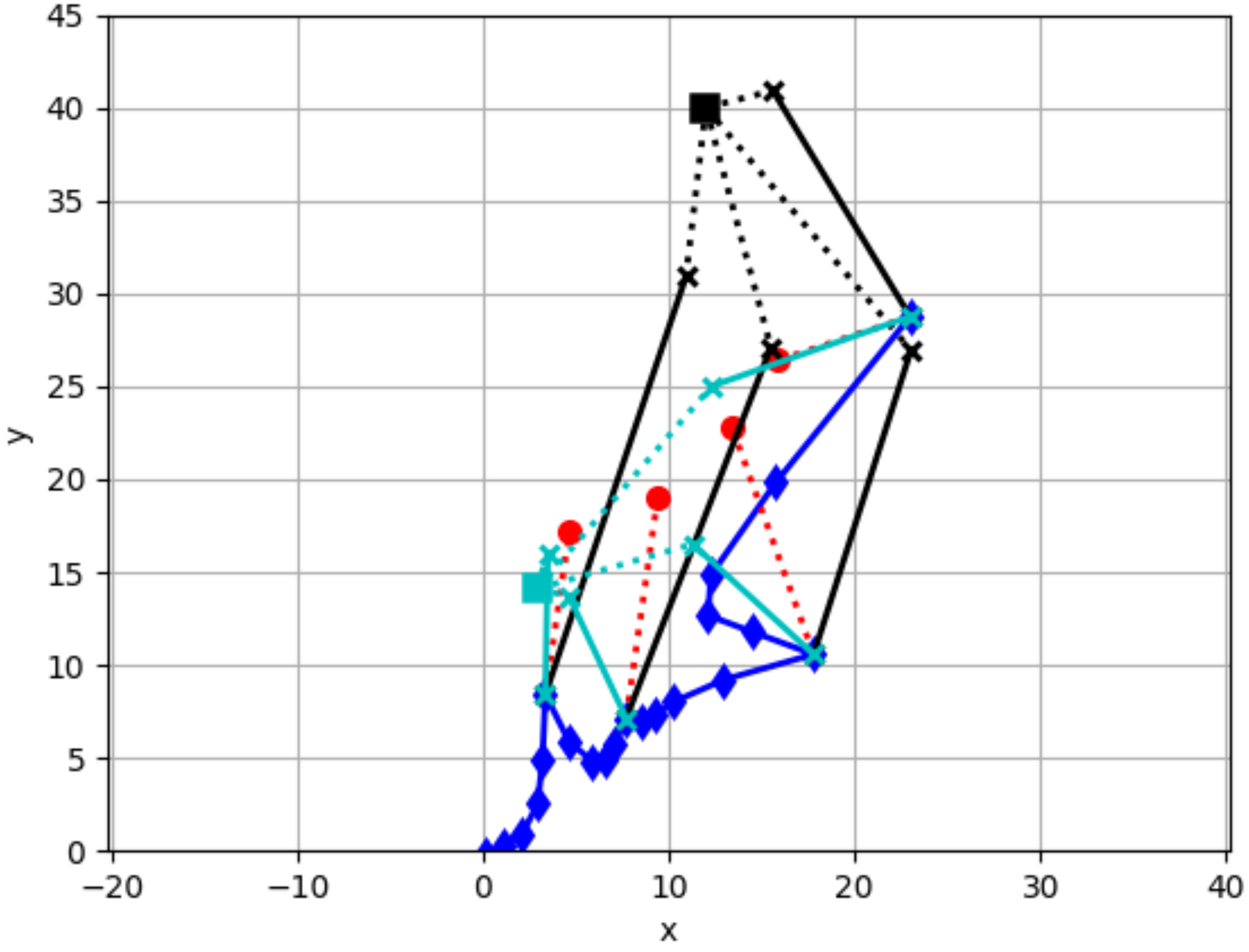
I can ignore the old stuff and solve it again but there is more efficient way

Optimization in SE(2) manifold trajectory length 21

noise:

- odom 0.2m / 0.2rad
- markers 0.3m / 0.3rad

Optimization in SE(2) manifold trajectory length 21



absolute marker

relative marker

odometry

some optimised poses

some ground truth poses (not used in optimisation)

noise:

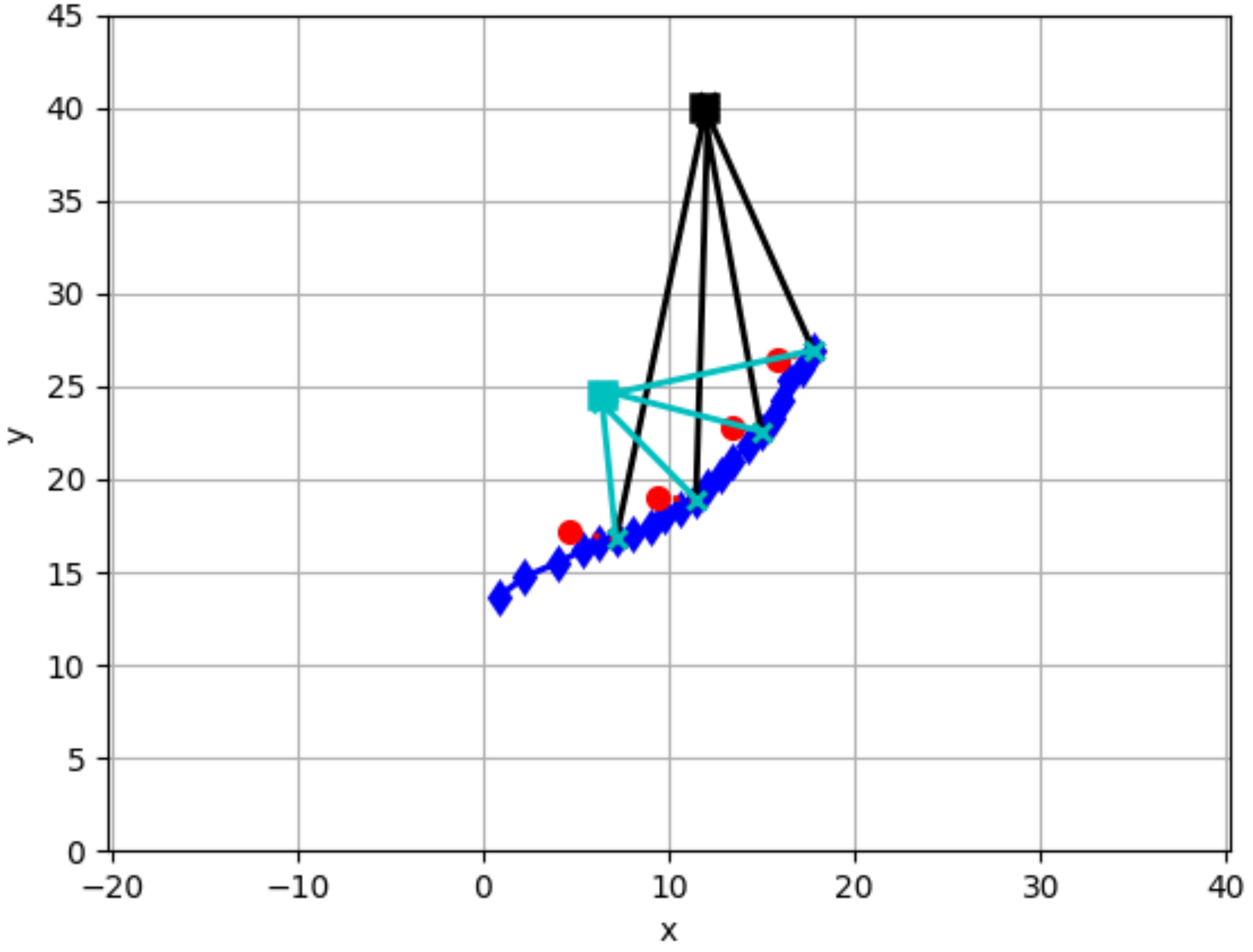
- odom 0.2m / 0.2rad
- markers 0.3m / 0.3rad

Optimization in SE(2) manifold trajectory length 21

noise:

- odom 0.2m / 0.2rad
- markers 0.3m / 0.3rad

Optimization in SE(2) manifold trajectory length 21

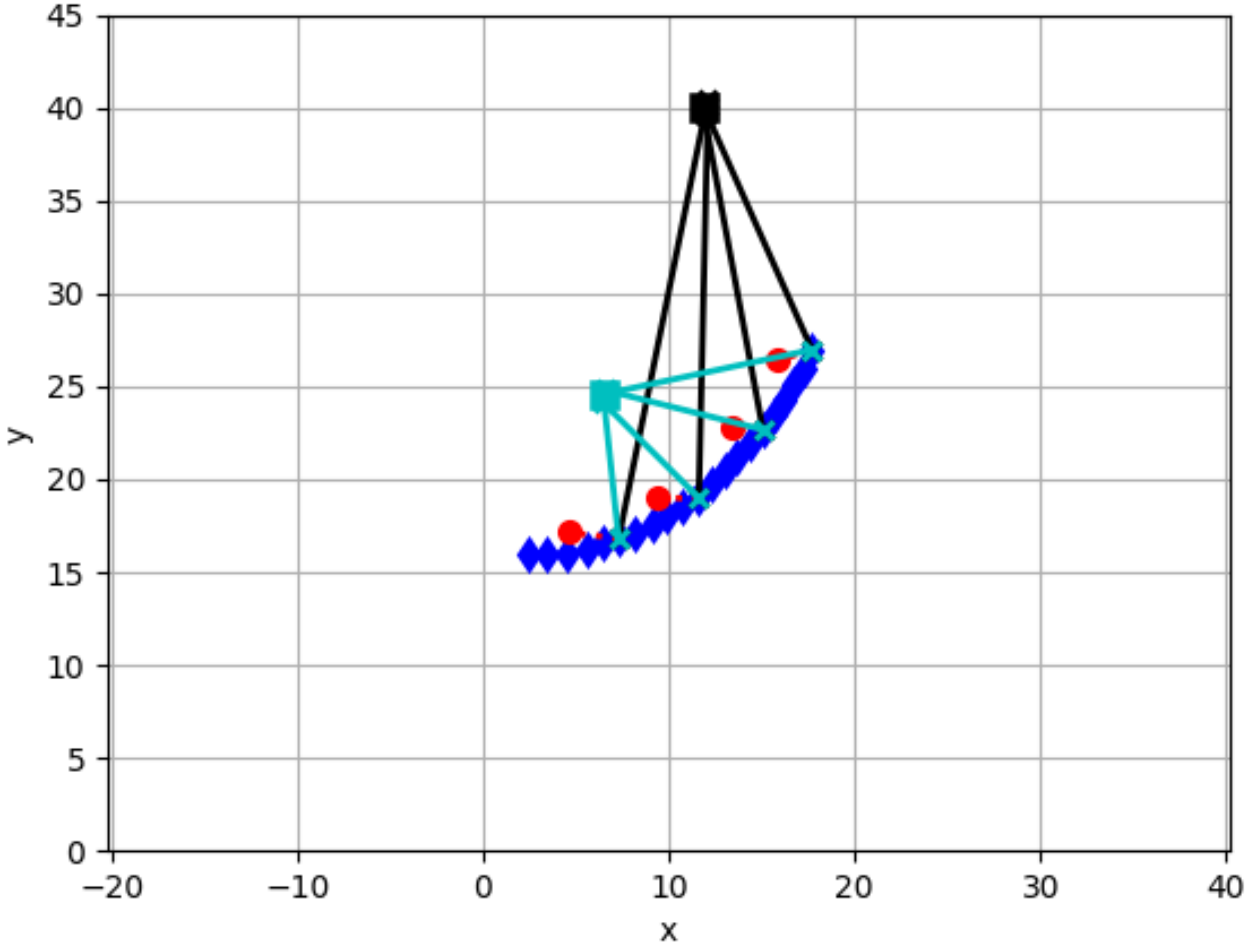


- absolute marker
- relative marker
- odometry
- some optimised poses
- some ground truth poses (not used in optimisation)

noise:

- odom 0.2m / 0.2rad
- markers 0.3m / 0.3rad

Optimization in SE(2) manifold trajectory length 21



- absolute marker
- relative marker
- odometry
- some optimised poses
- some ground truth poses (not used in optimisation)

noise:

- odom 0.2m / 0.2rad
- markers 0.3m / 0.3rad

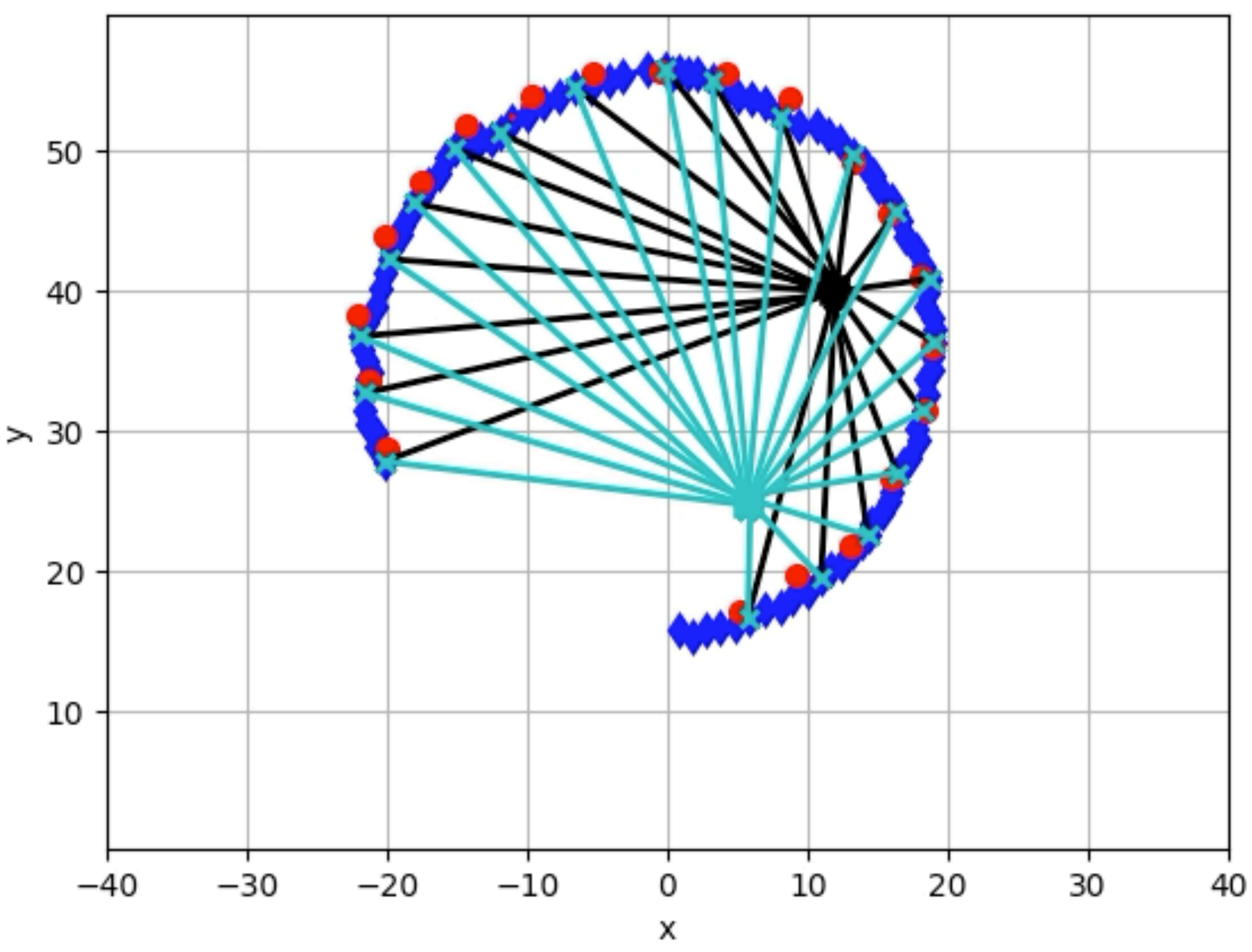
Optimization in SE(2) manifold trajectory length 101

What will break it????

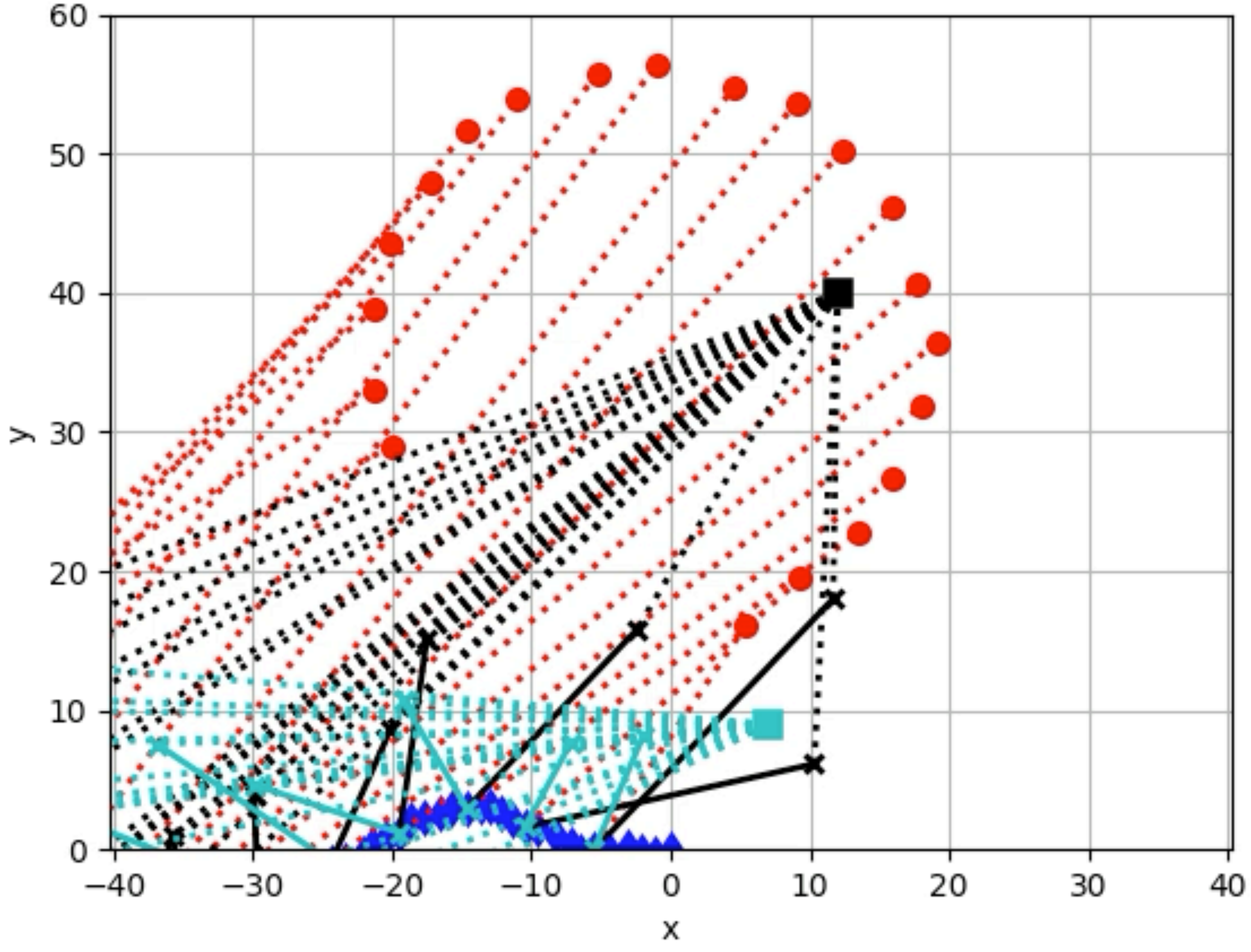
Optimization in SE(2) manifold trajectory length 101

noise 0.5
c_odom=0.2
c_ma=1
c_mr=1

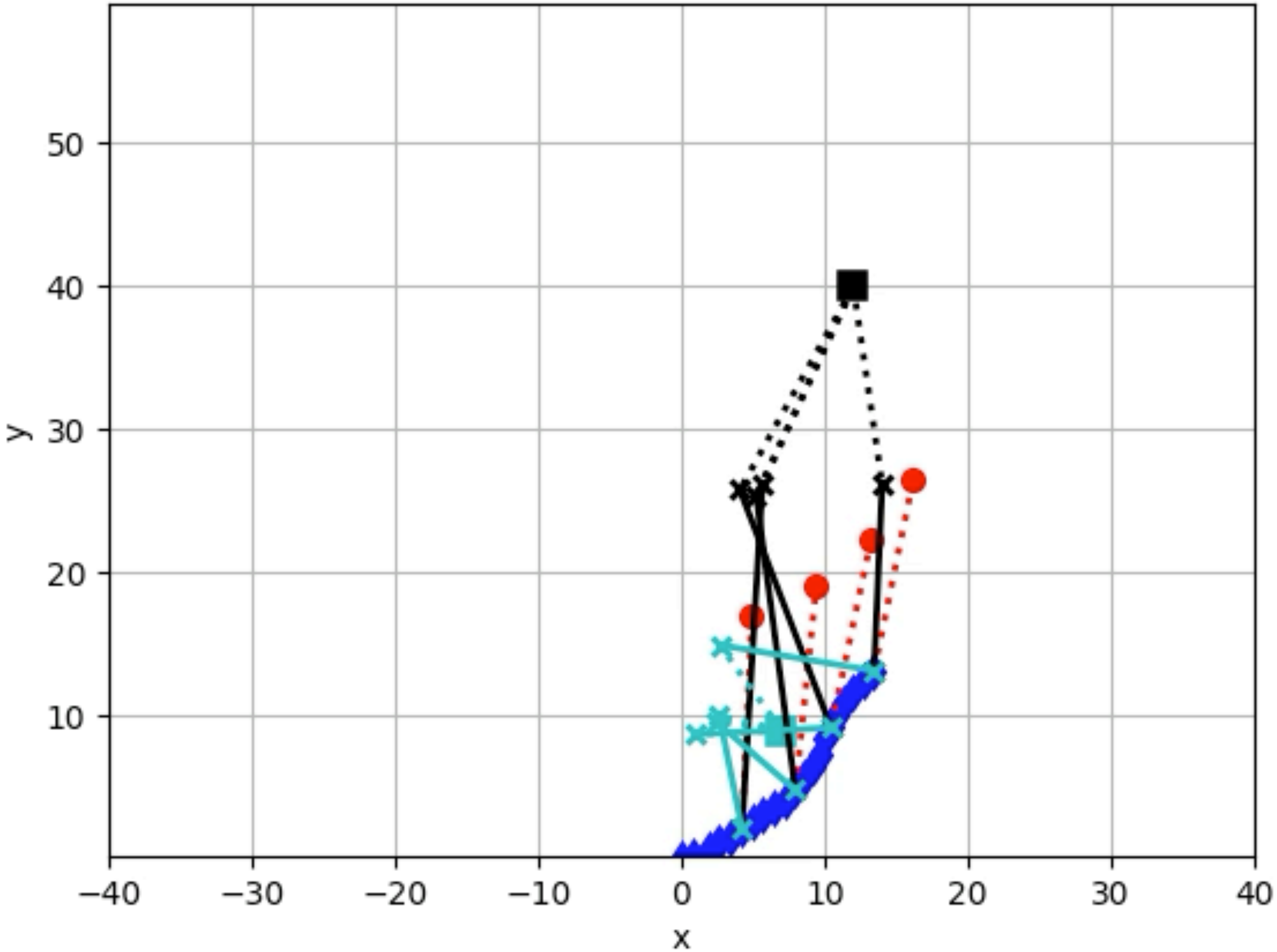
odom/marker ini



adversarial ini



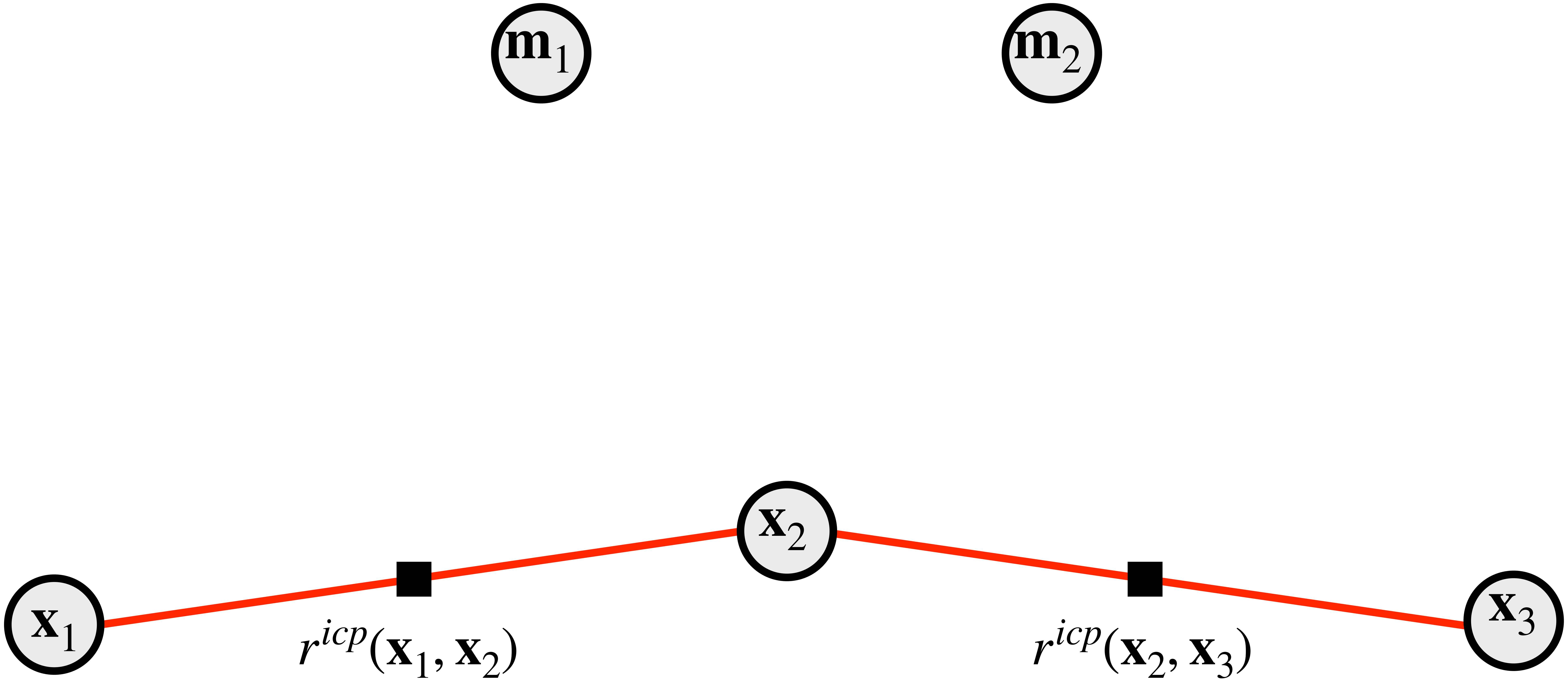
Optimization in SE(2) manifold trajectory length 401
successive optimization with incoming measurements
noise_markers = 0.5 noise_odom = 0.1



Optimization

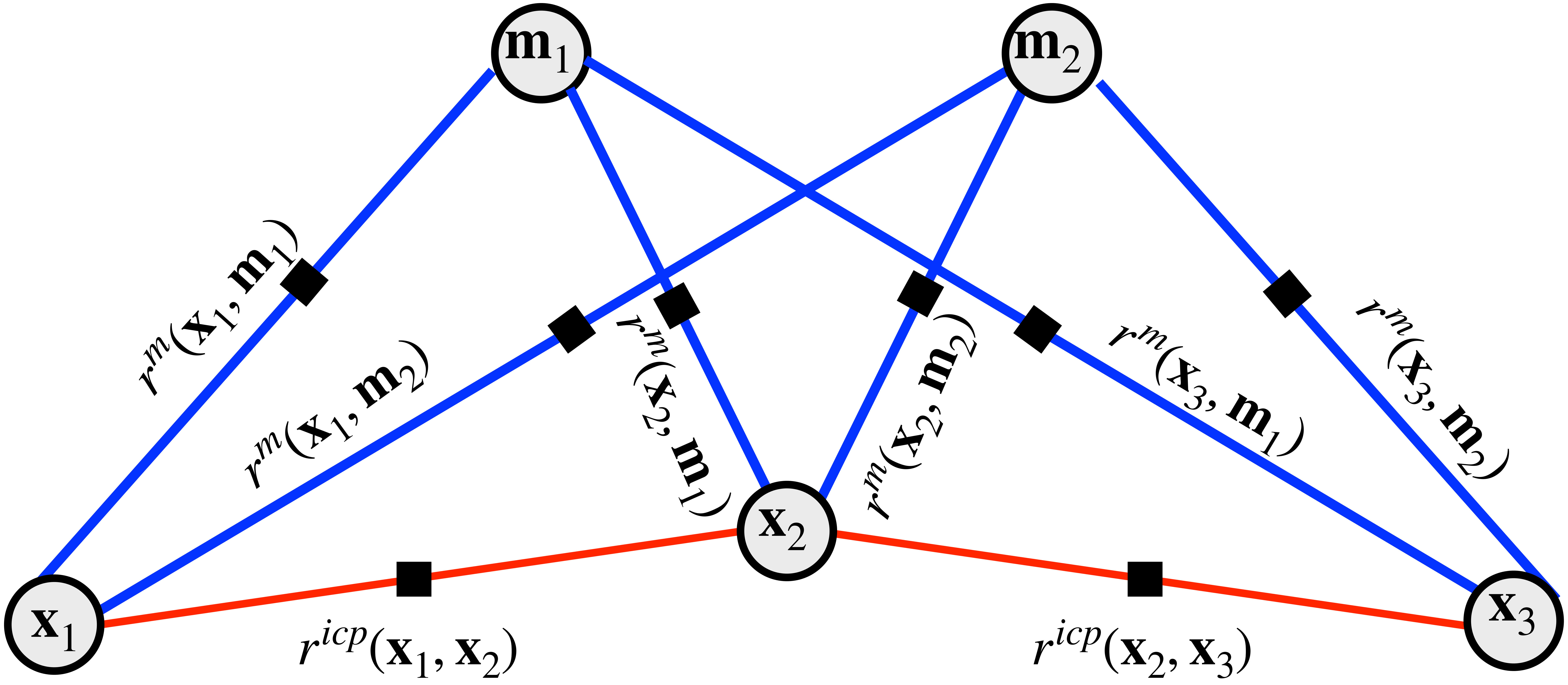
HW4 factorgraph

$$
= \arg\min_{\substack{\mathbf{x}_1 \ldots \mathbf{x}_T \\ \mathbf{m}_1 \ldots \mathbf{m}_J}} \overbrace{\sum_t \underbrace{\|\mathrm{w2r}(\mathbf{x}_{t+1}, \mathbf{x}_t) - \mathbf{z}_t^{icp}\|^2_{\Sigma_t^v}}_{r^{icp}(\mathbf{x}_t, \mathbf{x}_{t+1})}}^{\text{ICP odometry}} + \overbrace{\sum_{t,j} \underbrace{\|\mathrm{w2r}(\mathbf{m}_j, \mathbf{x}_t) - \mathbf{z}_t^{m_j}\|^2_{\Sigma_t^{m_j}}}_{r^m(\mathbf{x}_t, \mathbf{m}_j)}}^{\text{3D marker(s)}}
$$



HW4 factorgraph

$$= \arg\min_{\substack{\mathbf{x}_1\ldots\mathbf{x}_T \\ \mathbf{m}_1\ldots\mathbf{m}_J}} \overbrace{\underbrace{\sum_t \|\mathrm{w2r}(\mathbf{x}_{t+1}, \mathbf{x}_t) - \mathbf{z}_t^{icp}\|^2_{\Sigma_t^v}}_{r^{icp}(\mathbf{x}_t, \mathbf{x}_{t+1})}}^{\text{ICP odometry}} + \overbrace{\underbrace{\sum_{t,j} \|\mathrm{w2r}(\mathbf{m}_j, \mathbf{x}_t) - \mathbf{z}_t^{m_j}\|^2_{\Sigma_t^{m_j}}}_{r^{m}(\mathbf{x}_t, \mathbf{m}_j)}}^{\text{3D marker(s)}}$$



HW4 factorgraph

ICP odometry

3D marker(s)

$$= \arg\min_{\substack{\mathbf{x}_1\ldots\mathbf{x}_T\\ \mathbf{m}_1\ldots\mathbf{m}_J}} \underbrace{\sum_t \|\mathrm{w2r}(\mathbf{x}_{t+1},\mathbf{x}_t) - \mathbf{z}_t^{icp}\|^2_{\Sigma_t^v}}_{r^{icp}(\mathbf{x}_t,\mathbf{x}_{t+1})} + \underbrace{\sum_{t,j} \|\mathrm{w2r}(\mathbf{m}_j,\mathbf{x}_t) - \mathbf{z}_t^{m_j}\|^2_{\Sigma_t^{m_j}}}_{r^m(\mathbf{x}_t,\mathbf{m}_j)}$$

$$r^m(\mathbf{x}_t,\mathbf{m}_j) = \begin{bmatrix} +\cos\theta_t\cdot(m_j^x - x_t) + \sin\theta_t\cdot(m_j^y - y_t) - z_t^{m_j,x} \\ -\sin\theta_t\cdot(m_j^x - x_t) + \cos\theta_t\cdot(m_j^y - y_t) - z_t^{m_j,y} \\ m_j^\theta - \theta_t - z_t^{m_j,\theta} \end{bmatrix}$$

$$r^{icp}(\mathbf{x}_t,\mathbf{x}_{t+1}) = \begin{bmatrix} +\cos\theta_t\cdot(x_{t+1} - x_t) + \sin\theta_t\cdot(y_{t+1} - y_t) - z_t^{icp,x} \\ -\sin\theta_t\cdot(x_{t+1} - x_t) + \cos\theta_t\cdot(y_{t+1} - y_t) - z_t^{icp,y} \\ \theta_{t+1} - \theta_t - z_t^{icp,\theta} \end{bmatrix}$$

$$r^m(\mathbf{x}_t, \mathbf{m}_j) = \begin{bmatrix} +\cos\theta_t \cdot (m_j^x - x_t) + \sin\theta_t \cdot (m_j^y - y_t) - z_t^{m_j,x} \\ -\sin\theta_t \cdot (m_j^x - x_t) + \cos\theta_t \cdot (m_j^y - y_t) - z_t^{m_j,y} \\ m_j^\theta - \theta_t - z_t^{m_j,\theta} \end{bmatrix}$$

$$\mathbf{J}_x^m = \frac{\partial r^m(\mathbf{x}_t, \mathbf{m}_j)}{\partial \mathbf{x}_t} = \begin{array}{c} \begin{array}{ccc} \dfrac{\partial}{\partial x_t} & \dfrac{\partial}{\partial y_t} & \dfrac{\partial}{\partial \theta_t} \end{array} \\ \begin{bmatrix} -\cos\theta_t & -\sin\theta_t & -\sin\theta_t \cdot (m_j^x - x_t) + \cos\theta_t \cdot (m_j^y - y_t) \\ +\sin\theta_t & -\cos\theta_t & -\cos\theta_t \cdot (m_j^x - x_t) - \sin\theta_t \cdot (m_j^y - y_t) \\ 0 & 0 & -1 \end{bmatrix} \end{array}$$

$$\mathbf{J}_m^m = \frac{\partial r^m(\mathbf{x}_t, \mathbf{m}_j)}{\partial \mathbf{m}_j} = \begin{array}{c} \begin{array}{ccc} \dfrac{\partial}{\partial m_j^x} & \dfrac{\partial}{\partial m_j^y} & \dfrac{\partial}{\partial m_j^\theta} \end{array} \\ \begin{bmatrix} \cos\theta_t & \sin\theta_t & 0 \\ -\sin\theta_t & \cos\theta_t & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{array}$$

HW4 factorgraph

$$= \arg\min_{\substack{\mathbf{x}_1\ldots\mathbf{x}_T \\ \mathbf{m}_1\ldots\mathbf{m}_J}} \underbrace{\overbrace{\sum_t \|\mathrm{w2r}(\mathbf{x}_{t+1}, \mathbf{x}_t) - \mathbf{z}_t^{icp}\|^2_{\Sigma_t^y}}^{\text{ICP odometry}}}_{r^{icp}(\mathbf{x}_t, \mathbf{x}_{t+1})} + \underbrace{\overbrace{\sum_{t,j} \|\mathrm{w2r}(\mathbf{m}_j, \mathbf{x}_t) - \mathbf{z}_t^{m_j}\|^2_{\Sigma_t^{m_j}}}^{\text{3D marker(s)}}}_{r^m(\mathbf{x}_t, \mathbf{m}_j)}$$

$$r^m(\mathbf{x}_t, \mathbf{m}_j) = \begin{bmatrix} +\cos\theta_t \cdot (m_j^x - x_t) + \sin\theta_t \cdot (m_j^y - y_t) - z_t^{m_j,x} \\ -\sin\theta_t \cdot (m_j^x - x_t) + \cos\theta_t \cdot (m_j^y - y_t) - z_t^{m_j,y} \\ m_j^\theta - \theta_t - z_t^{m_j,\theta} \end{bmatrix}$$

$$r^{icp}(\mathbf{x}_t, \mathbf{x}_{t+1}) = \begin{bmatrix} +\cos\theta_t \cdot (x_{t+1} - x_t) + \sin\theta_t \cdot (y_{t+1} - y_t) - z_t^{icp,x} \\ -\sin\theta_t \cdot (x_{t+1} - x_t) + \cos\theta_t \cdot (y_{t+1} - y_t) - z_t^{icp,y} \\ \theta_{t+1} - \theta_t - z_t^{icp,\theta} \end{bmatrix}$$

$$
r^{icp}(\mathbf{x}_t, \mathbf{x}_{t+1}) = \begin{bmatrix} +\cos\theta_t \cdot (x_{t+1} - x_t) + \sin\theta_t \cdot (y_{t+1} - y_t) - z_t^{icp,x} \\ -\sin\theta_t \cdot (x_{t+1} - x_t) + \cos\theta_t \cdot (y_{t+1} - y_t) - z_t^{icp,y} \\ \theta_{t+1} - \theta_t - z_t^{icp,\theta} \end{bmatrix}
$$

$$
\mathbf{J}_{x_t}^{icp} = \frac{\partial r^{icp}(\mathbf{x}_t, \mathbf{x}_{t+1})}{\partial \mathbf{x}_t} = \begin{array}{c} \begin{array}{ccc} \dfrac{\partial}{\partial x_t} & \dfrac{\partial}{\partial y_t} & \dfrac{\partial}{\partial \theta_t} \end{array} \\ \begin{bmatrix} -\cos\theta_t & -\sin\theta_t & -\sin\theta_t \cdot (x_{t+1} - x_t) + \cos\theta_t \cdot (y_{t+1} - y_t) \\ +\sin\theta_t & -\cos\theta_t & -\cos\theta_t \cdot (x_{t+1} - x_t) - \sin\theta_t \cdot (y_{t+1} - y_t) \\ 0 & 0 & -1 \end{bmatrix} \end{array}
$$

$$
\mathbf{J}_{x_{t+1}}^{icp} = \frac{\partial r^{icp}(\mathbf{x}_t, \mathbf{x}_{t+1})}{\partial \mathbf{x}_{t+1}} = \begin{array}{c} \begin{array}{ccc} \dfrac{\partial}{\partial x_{t+1}} & \dfrac{\partial}{\partial y_{t+1}} & \dfrac{\partial}{\partial \theta_{t+1}} \end{array} \\ \begin{bmatrix} \cos\theta_t & \sin\theta_t & 0 \\ -\sin\theta_t & \cos\theta_t & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{array}
$$

Optimization

```
scipy.optimize.least_squares(fun, x0, jac, method='lm')
```

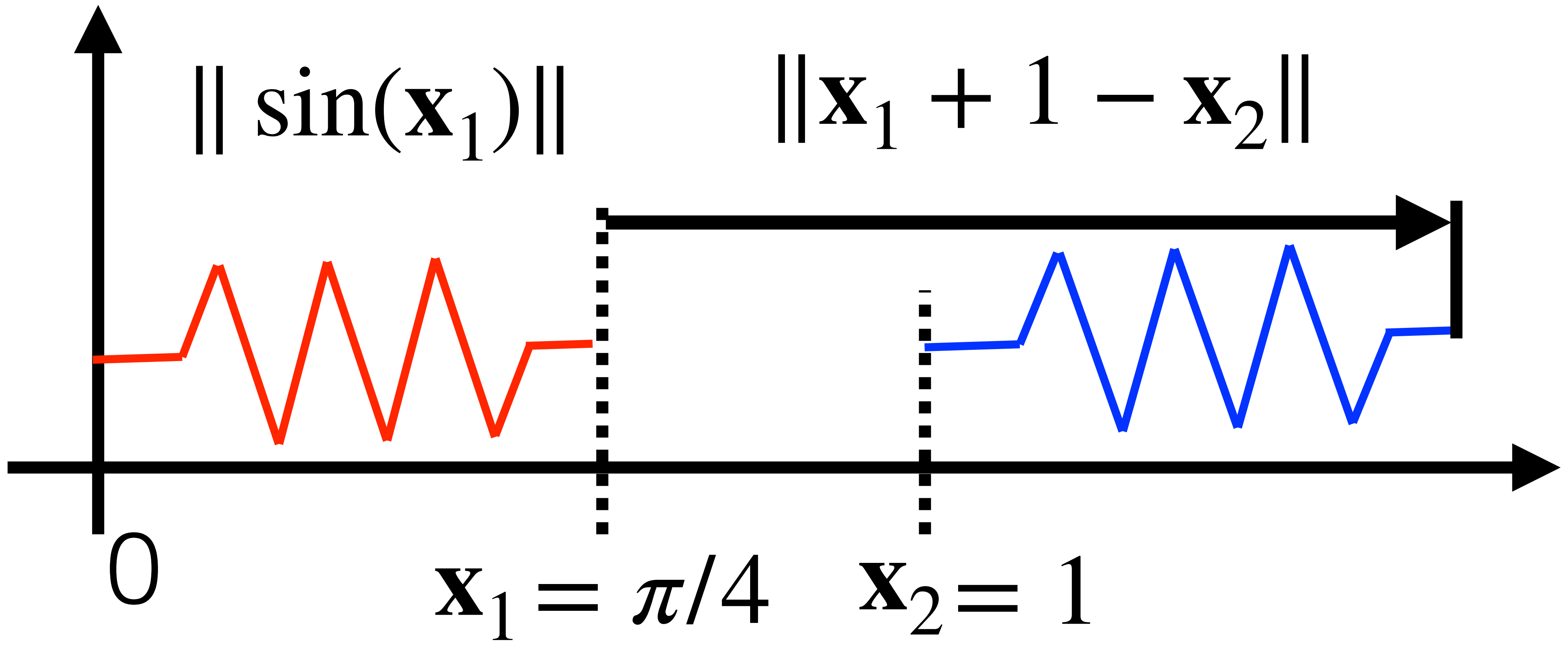
Optimization **time** grows fast with:

- problem **dimensionality** (e.g. DOFxT+M)
- number of **residual terms** (e.g. number of measurements)

In practise you introduce **simplifications**:

- use **sparse matrix** to represent to represent inherently sparse Jacobian
- optimize only relevant **sub-graph** when new measurement comes (Givens rot.)
- **pre-integrate** some factor (e.g. sum up odometry measurements over 0.5s)
- **sparsification** of old factor graph
- to tackle the real-time requirements **frontend** and **backend** optimizers used
- general FG solvers such as gtsam with iSAM/iSAM2 optimizers available:

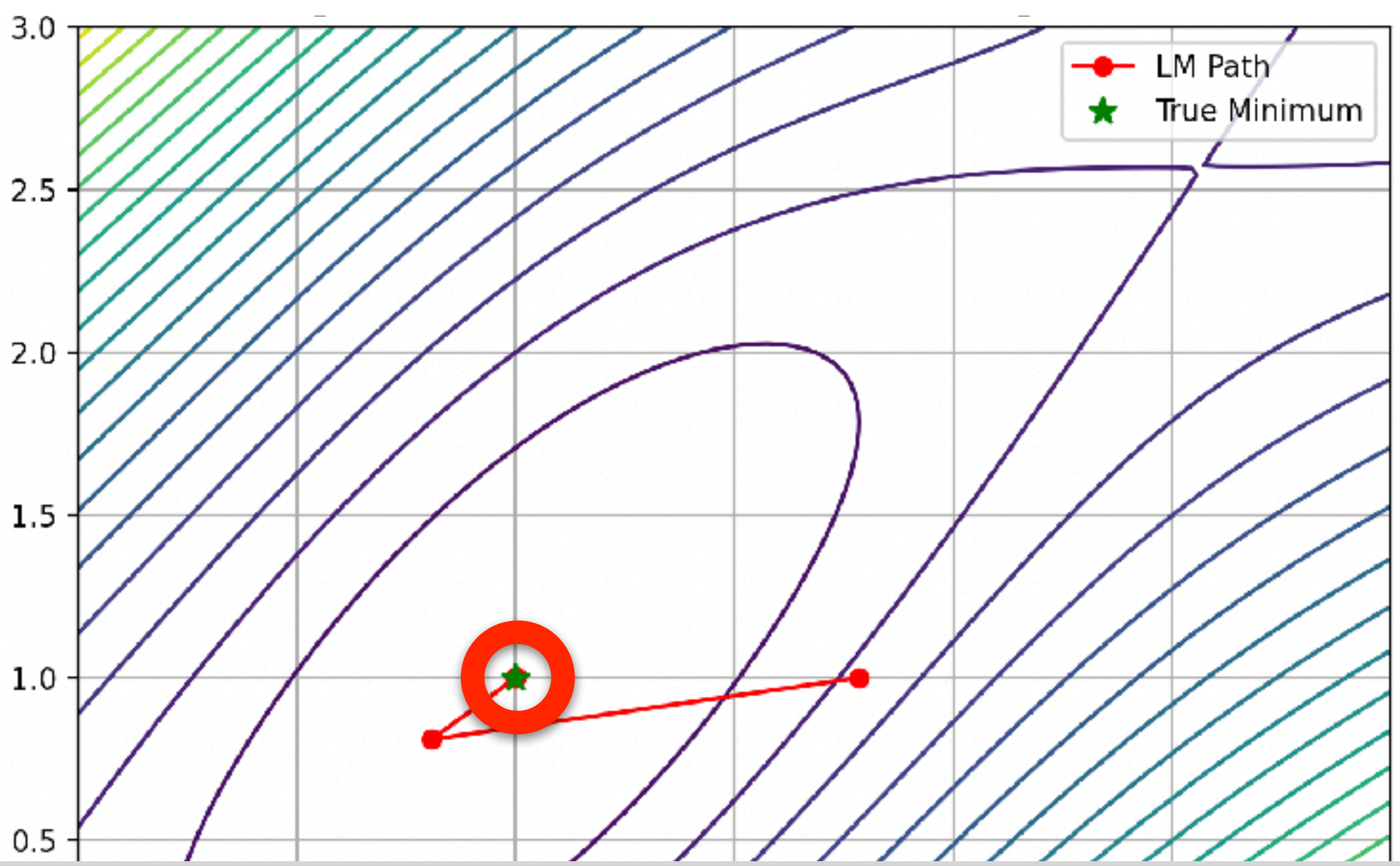
Let's try it on 2D factorgraph problem



$$\Delta \mathbf{x} = \arg\min_{\mathbf{x}_1, \mathbf{x}_2} \left(\sin \mathbf{x}_1\right)^2 + \left(\mathbf{x}_1 + 1 - \mathbf{x}_2\right)^2$$

$$\mathbf{f} = \begin{bmatrix} \sin(\pi/4) \\ \pi/4 + 1 - 1 \end{bmatrix}$$

$$\mathbf{J} = \begin{bmatrix} \cos(\pi/4) & 0 \\ -1 & 1 \end{bmatrix}$$




```
import gtsam
initial.insert(x1, np.pi/4)
initial.insert(x2, 1.0)

graph.add(SinFactor(x1, noise_model))
graph.add(gtsam.BetweenFactorDouble(x1, x2, 1.0, noise_model))

# Update iSAM2 and optimize incrementally
isam.update(graph, initial)
result = isam.calculateEstimate()
```