

Deck 04 · Optimization I: descent, Newton, Gauss-Newton, and the damped inverse-kinematics step

3.1 Descent, step size, and the safety check

3.2 Newton, Gauss-Newton, and the damped step

Özgür S. Ögüz · Bilkent University · Fall 2026

Given φ and $\mathbf{J}$, how do we find $\mathbf{q}$ with $\varphi(\mathbf{q}) = \mathbf{y}^*$?

When does the obvious full step fail, what check protects against it, and what better step replaces it?

Where Deck 03's bridge left us

quantity	value
target $\mathbf{y}^*$	(0.5, 0.2) m
start $\mathbf{q}^{(0)}$	(30°, 45°)
tip $\varphi(\mathbf{q}^{(0)})$	(0.4241, 0.4898) m
$\mathbf{e} = \mathbf{y}^* - \varphi(\mathbf{q}^{(0)})$	(0.0759, -0.2898) m
step solving $\mathbf{J}\Delta\mathbf{q} = \mathbf{e}$	(-52.7°, 74.1°)
miss after that step	0.17 m

Left open

- How far to trust the step
- A $3 \times 7 \mathbf{J}$ (the Panda)
- What **closest** means when no $\mathbf{q}$ reaches $\mathbf{y}^*$

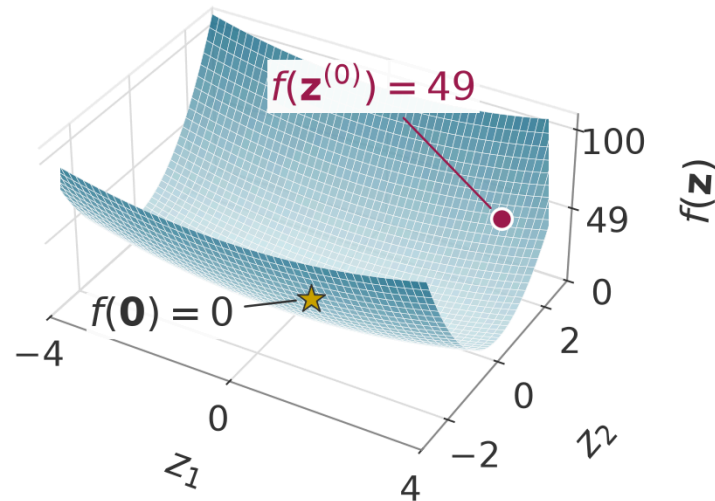
A scalar cost and methods that walk downhill on it answer all three.

$$\min_{\mathbf{q}} f(\mathbf{q}) = \frac{1}{2} \|\varphi(\mathbf{q}) - \mathbf{y}^*\|^2, \quad \mathbf{r}(\mathbf{q}) := \varphi(\mathbf{q}) - \mathbf{y}^*, \quad \mathbf{r}(\mathbf{q}^{(0)}) = -\mathbf{e}$$

See the cost as a surface

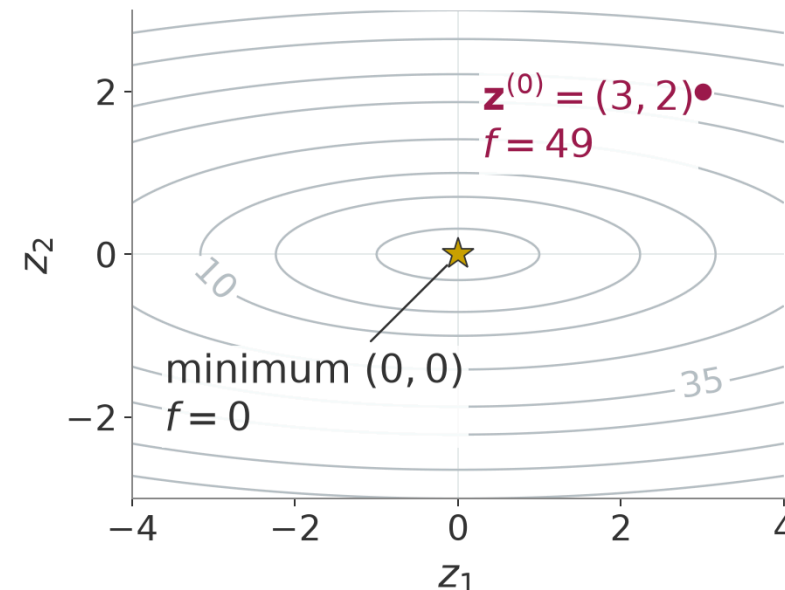
$$f(\mathbf{z}) = z_1^2 + 10z_2^2, \quad \mathbf{z} \in \mathbb{R}^2, \quad \mathbf{z}^{(0)} = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

Height gives the cost



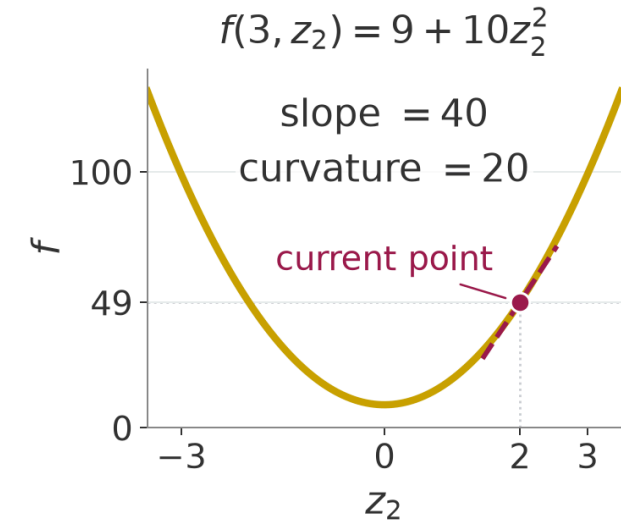
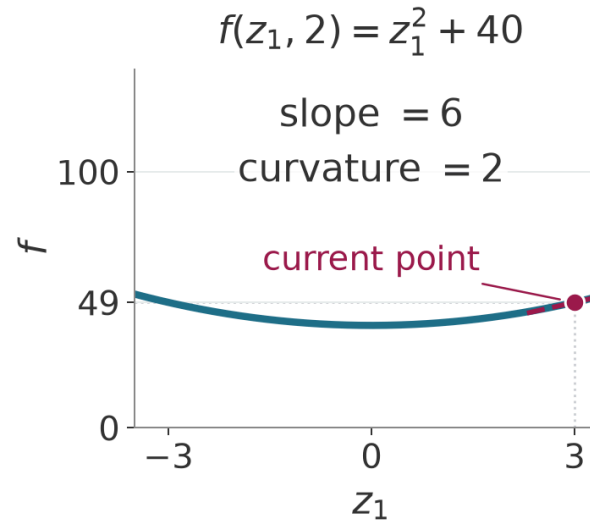
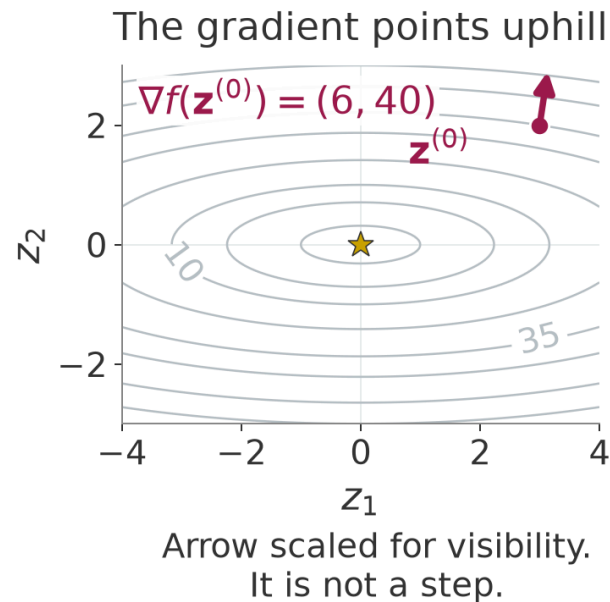
Current cost: $f(3, 2) = 3^2 + 10(2^2) = 49$

Each contour has one cost



Minimum: $(0, 0)$, cost 0 · **Contours:** equal cost

Read slope and curvature at the same point



Both slices use the same axis scales.

Gradient: the two slopes, stacked; points uphill

$$\nabla f(\mathbf{z}) = \begin{bmatrix} 2z_1 \\ 20z_2 \end{bmatrix}, \quad \nabla f(3, 2) = \begin{bmatrix} 6 \\ 40 \end{bmatrix}$$

Hessian: changes of those slopes

$$H_{ij} = \frac{\partial^2 f}{\partial z_i \partial z_j}, \quad \mathbf{H} = \begin{bmatrix} 2 & 0 \\ 0 & 20 \end{bmatrix}$$

$\mathbf{H} \succ 0$ (**positive definite**): upward curvature in every direction

The local model, and what "local" costs

$$\min_{\mathbf{z} \in \mathbb{R}^d} f(\mathbf{z}), \quad f(\mathbf{z} + \boldsymbol{\delta}) \approx f(\mathbf{z}) + \nabla f^\top \boldsymbol{\delta} + \frac{1}{2} \boldsymbol{\delta}^\top \mathbf{H} \boldsymbol{\delta}$$

- Deck 03 linearized a vector map, $\varphi(\mathbf{q} + \Delta\mathbf{q}) \approx \varphi(\mathbf{q}) + \mathbf{J}\Delta\mathbf{q}$; this is the same expansion for a scalar, kept one term further, with the step $\Delta\mathbf{q}$ written $\boldsymbol{\delta}$
- Today: $\mathbf{z} = \mathbf{q}$, the n joint angles
- Week 4: $\mathbf{z} = \mathbf{q}_{0:T}$, a whole trajectory
- **Local search** is like a walker on a hillside in fog, sensing nearby ground. Different starts can reach different valleys; a run may stop at a saddle or fail to converge. No global-minimum certificate

Two decisions per step

$$\mathbf{z} \leftarrow \mathbf{z} + \alpha \mathbf{d}$$

- **Where to move:** downhill direction $\mathbf{d}$
- **How far to move:** scale $\mathbf{d}$ by $\alpha > 0$
- Gradient descent: $\mathbf{d} = -\nabla f(\mathbf{z})$
- Newton: $\mathbf{d} = -\mathbf{H}^{-1} \nabla f(\mathbf{z})$ (slide 17 derives it)

Wanted from the method

- **monotone:** f never increases
- **convergent:** $\|\nabla f\| \rightarrow 0$ (conditions in the notes)
- **robust:** rescaling $\mathbf{z}$ does not slow it

Compute a gradient-descent step

$$f(\mathbf{z}) = z_1^2 + 10z_2^2, \quad \nabla f(\mathbf{z}) = (2z_1, 20z_2)$$

1. Subtract the scaled slope.

$$\begin{aligned} z_1 &\leftarrow z_1 - \alpha(2z_1) \\ &= (1 - 2\alpha)z_1, \end{aligned}$$

$$\begin{aligned} z_2 &\leftarrow z_2 - \alpha(20z_2) \\ &= (1 - 20\alpha)z_2. \end{aligned}$$

2. Compare step sizes from (3, 2).

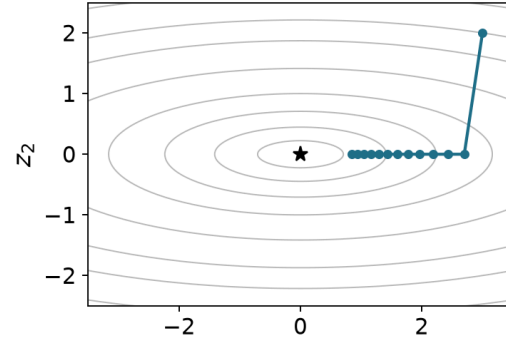
α	factors on (z_1, z_2)	next point
0.05	(0.9, 0)	(2.7, 0)
0.5	(0, -9)	(0, -18)

- 0.05: z_2 at zero; z_1 closer
- 0.5: z_1 at zero; z_2 overshoots

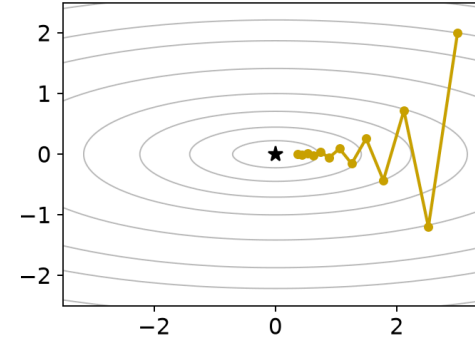
No single α takes both to zero in one step.

Three step sizes, and Newton's one step

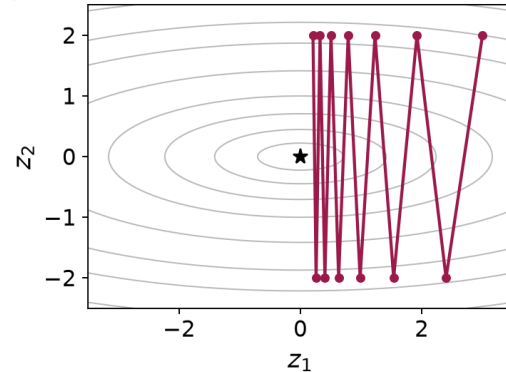
gradient descent, $\alpha = 0.05$: factors (0.90, 0.00)



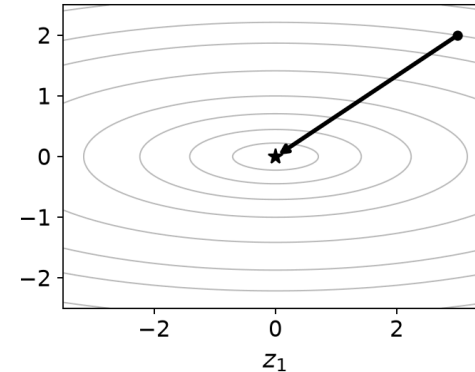
gradient descent, $\alpha = 0.08$: factors (0.84, -0.60)



gradient descent, $\alpha = 0.1$: factors (0.80, -1.00)



Newton: $z - H^{-1}\nabla f$, one step, exactly



$f(\mathbf{z}) = z_1^2 + 10z_2^2$ from $(3, 2)$, twelve steps: larger α speeds z_1 but can make z_2 oscillate

Balance the two coordinate updates

1. Compare the curvatures (eigenvalues of $\mathbf{H}$).

$$\lambda_{\min} = 2, \quad \lambda_{\max} = 20, \quad \text{cond}(\mathbf{H}) = \frac{\lambda_{\max}}{\lambda_{\min}} = \frac{20}{2} = 10$$

Condition number: steepest / flattest curvature

2. Make the larger factor magnitude as small as possible.

Equal factor magnitudes, opposite signs

$$1 - 2\alpha = -(1 - 20\alpha) \quad \Rightarrow \quad 2 = 22\alpha \quad \Rightarrow \quad \alpha^* = \frac{1}{11} \approx 0.0909$$

3. Repeat this fixed step.

$$z_1 \leftarrow \frac{9}{11} z_1, \quad z_2 \leftarrow -\frac{9}{11} z_2 \quad \Rightarrow \quad f_{\text{next}} = \frac{81}{121} f \approx 0.669 f$$

From cost 49: **45 steps** to cost below 10^{-6} .

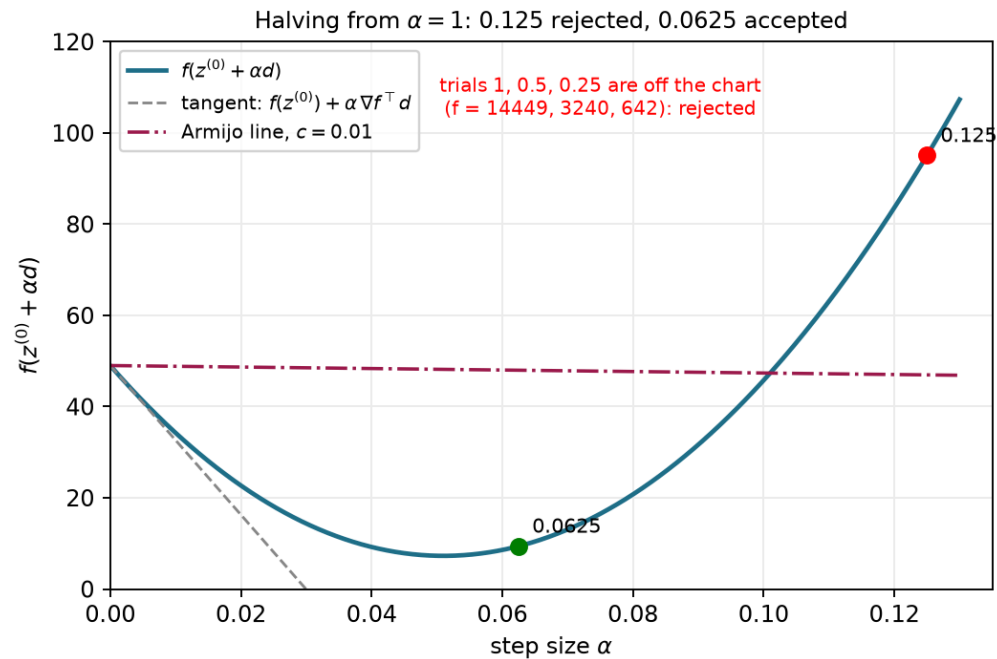
Larger curvature mismatch: slower progress in the slowest direction

Armijo's check chooses the step size

$$f(\mathbf{z} + \alpha \mathbf{d}) \leq f(\mathbf{z}) + c \alpha \nabla f(\mathbf{z})^\top \mathbf{d}, \quad c = 0.01$$

- The best fixed α needs $\lambda_{\min}$ and $\lambda_{\max}$, which a real cost does not tell you and which change with $\mathbf{z}$
- Accept the step only if f falls by at least the fraction c of the drop that the tangent line predicts
- If not: **halve α and try again**. If yes: take the step, grow α by 1.2 for next time.
- For a smooth cost and a strict descent direction, unlimited backtracking terminates; accepted steps make descent **monotone**

Backtracking at $\mathbf{z}^{(0)} = (3, 2)$



- $\mathbf{d} = -\nabla f = (-6, -40)$
- threshold = $49 - 16.36 \alpha$

α	$f(\mathbf{z}^{(0)} + \alpha \mathbf{d})$	threshold	
1	14 449	32.6	reject
0.5	3 240	40.8	reject
0.25	642	44.9	reject
0.125	95.1	47.0	reject
0.0625	9.4	48.0	accept

step	1	2	3	4	5	6
accepted α	0.0625	0.075	0.09	0.108	0.13	0.078

The search adapts α at each point.

The algorithm, and what is left out

```
 $\alpha \leftarrow 1$ 
repeat
   $d \leftarrow -\nabla f(z)$ 
  while  $f(z + \alpha d) > f(z) + c \alpha \nabla f(z)^\top d$ :           # Armijo backtracking
    if  $\alpha \leq \text{floor}$ : stop, report "stalled"           # floor  $10^{-6}$ 
     $\alpha \leftarrow \alpha / 2$ 
   $s \leftarrow \alpha d$ ;  $z \leftarrow z + s$                  # the accepted step
   $\alpha \leftarrow 1.2 \alpha$                              # grow for the next trial
until  $|s| < \text{tolerance}$  or  $|\nabla f(z)| < \text{tolerance}$ 
```

A short accepted step means slow progress, not proven stationarity; a small gradient means approximate stationarity. Say which test stopped the run (slide 28).

In the backup slides at the end of this deck: Wolfe's curvature condition and bracket-and-zoom; quasi-Newton (BFGS, L-BFGS); conjugate gradients.

Live demo: the optimization explorer

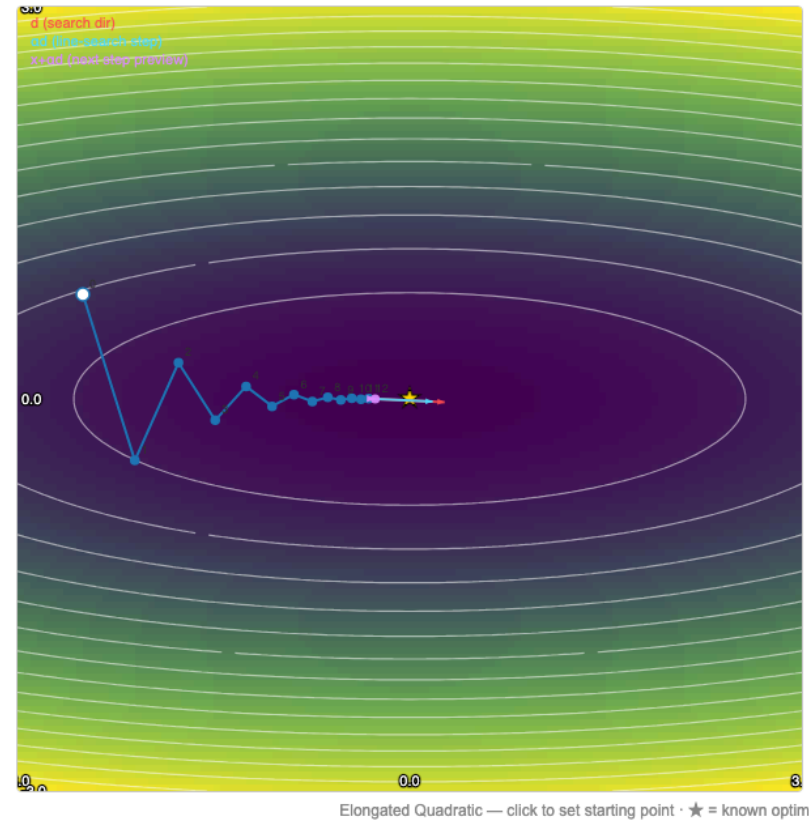
1 • Line search Off, gradient descent

slider α	before Step
0.050	try
0.079	try
0.100	try
0.110	predict, then run

2 • Line search Armijo, slider $\alpha = 1.000$

Compare "α used" with slide 11

3 • Line search Off, method **Newton**: one Step



Back to the arm: one gradient step

$$f(\mathbf{q}) = \frac{1}{2} \|\varphi(\mathbf{q}) - \mathbf{y}^*\|^2 \quad \Rightarrow \quad \nabla f(\mathbf{q}) = \mathbf{J}(\mathbf{q})^\top (\varphi(\mathbf{q}) - \mathbf{y}^*)$$

$\mathbf{q}^{(0)}$	$\mathbf{y}^*$	∇f (m ² /rad, checked by finite differences)
(30°, 45°)	(0.5, 0.2)	(0.160, 0.045)

	$\mathbf{q}$ (deg)	tip (m)	$\ \varphi - \mathbf{y}^*\ $
start	(30.0, 45.0)	(0.424, 0.490)	0.300 m
one step, $\alpha = 1$ accepted by Armijo	(20.8, 42.4)	(0.509, 0.410)	0.210 m

Stepping against this gradient is called **Jacobian-transpose inverse kinematics**. Armijo makes the step size safe.

Summary of 3.1

- Inverse kinematics becomes $\min_{\mathbf{q}} \frac{1}{2} \|\varphi(\mathbf{q}) - \mathbf{y}^*\|^2$, gradient $\mathbf{J}^\top (\varphi - \mathbf{y}^*)$: one gradient step took the tip from 30 cm to 21 cm off
- Gradient descent: the steepest curvature limits α
- Quadratic, fixed step: $\text{cond}(\mathbf{H})$ controls the worst-direction rate
- **Armijo backtracking**: choose α per step; require sufficient cost decrease

bowl	$\text{cond}(\mathbf{H})$	steps
today's	10	45
a narrow valley	100	443
the arm near its solution	8.2	56 (notes §4.2)

Break

Second half: use the curvature, then use the structure of the cost.

Newton uses curvature to choose the step

Fit a quadratic model around the current configuration:

$$m(\boldsymbol{\delta}) = f(\mathbf{z}) + \nabla f^\top \boldsymbol{\delta} + \frac{1}{2} \boldsymbol{\delta}^\top \mathbf{H} \boldsymbol{\delta}$$

If the model curves upward in every direction, its minimum gives the **Newton step**:

$$\mathbf{H} \boldsymbol{\delta} = -\nabla f$$

- For an upward-curving quadratic cost, one full step reaches the minimum
- For a nonlinear robot cost, check the actual result and rebuild the local model

Damping can correct an uphill Newton direction

A quadratic model can curve downward. Its stationary point need not minimize it.

Add enough positive curvature to make the model curve upward:

$$(\mathbf{H} + \rho \mathbf{I})\delta = -\nabla f$$

Live example: a cost with a maximum at the origin, starting at $(0.1, 0.1)$.

step	lands at	direction
Newton, $\rho = 0$	$(-0.002, -0.002)$	uphill, toward the maximum
damped, $\rho = 10$	$(0.165, 0.165)$	downhill

A downhill direction still needs an **actual-cost check**.

Gauss–Newton approximates Newton's curvature

Minimize the robot's squared tip error:

$$\min_{\mathbf{q}} f(\mathbf{q}) = \frac{1}{2} \|\mathbf{r}(\mathbf{q})\|^2, \quad \mathbf{r}(\mathbf{q}) := \varphi(\mathbf{q}) - \mathbf{y}^*$$

First derivative, then second derivative:

$$\nabla f = \mathbf{J}_r^\top \mathbf{r}$$

$$\mathbf{H} = \underbrace{\mathbf{J}_r^\top \mathbf{J}_r}_{\text{Gauss–Newton keeps this}} + \underbrace{\sum_j r_j \nabla^2 r_j}_{\text{Gauss–Newton drops this}}$$

- Small residuals often make the omitted curvature small
- Persistent residuals can make that curvature important

Gauss–Newton reduces the predicted error

Reuse differential IK's Jacobian: choose a **joint displacement** instead of a velocity.

Minimize the predicted squared error for one step:

$$\min_{\boldsymbol{\delta}} \frac{1}{2} \|\mathbf{r} + \mathbf{J}_r \boldsymbol{\delta}\|^2$$

Setting the model's gradient to zero gives the **Gauss–Newton step**:

$$(\mathbf{J}_r^\top \mathbf{J}_r) \boldsymbol{\delta} = -\mathbf{J}_r^\top \mathbf{r}$$

For IK, the Jacobian is the same, and the residual reverses Deck 03's displacement:

$$\mathbf{J}_r = \mathbf{J}, \quad \mathbf{r} = \varphi(\mathbf{q}) - \mathbf{y}^* = -\mathbf{e}$$

Damping makes the proposed step more cautious

$$(\mathbf{J}_r^\top \mathbf{J}_r + \rho \mathbf{I}) \boldsymbol{\delta} = -\mathbf{J}_r^\top \mathbf{r}$$

Same arm, target, and start, with slide 18's damping idea:

first step	undamped	damping $\rho = 0.05$
elbow change	74.1°	11.5°
actual tip error after the step	0.174 m	0.115 m

- Penalizes large joint changes; positive damping makes the system invertible
- With strong damping, curvature has less influence, and the update approaches a short negative-gradient step
- Fixed damping can slow progress, especially along weak task-motion directions

Let the solver adjust the damping

Levenberg–Marquardt (LM) adapts damping as the local model is rebuilt.

For a trial with a predicted cost decrease:

$$\eta_{\text{LM}} = \frac{\text{actual cost decrease}}{\text{predicted cost decrease}}$$

What the trial reveals	How the solver responds
Ratio near one or larger	Useful prediction; damping can decrease
Small positive ratio	Optimistic prediction; increase damping
Actual cost increases	Reject the step; retry with more damping

Still a local method: reaching the target is not guaranteed.

Inverse kinematics as least squares

The task objective

$$\min_{\mathbf{q}} \frac{1}{2} \|\mathbf{r}(\mathbf{q})\|_{\mathbf{K}}^2$$

$\mathbf{K}$ weights the task errors.

The model for one step

$$\min_{\boldsymbol{\delta}} \frac{1}{2} \|\mathbf{r} + \mathbf{J}\boldsymbol{\delta}\|_{\mathbf{K}}^2 + \frac{1}{2} \|\boldsymbol{\delta}\|_{\mathbf{D}}^2$$

$\mathbf{D}$ penalizes the joint change.

Here the residual and Jacobian are evaluated at the current configuration. For the task error:

$$\|\mathbf{r}\|_{\mathbf{K}}^2 = \mathbf{r}^\top \mathbf{K} \mathbf{r}$$

The step penalty discourages large updates from the current configuration. A posture cost favors staying near a fixed reference posture.

Damped Gauss–Newton for inverse kinematics

Solve for the joint change using the current error and Jacobian:

$$(\mathbf{J}^\top \mathbf{K} \mathbf{J} + \mathbf{D}) \boldsymbol{\delta} = -\mathbf{J}^\top \mathbf{K} \mathbf{r}$$

The same mapping, when the step matrix is invertible:

$$\mathbf{J}^\# = (\mathbf{J}^\top \mathbf{K} \mathbf{J} + \mathbf{D})^{-1} \mathbf{J}^\top \mathbf{K}, \quad \boldsymbol{\delta} = -\mathbf{J}^\# \mathbf{r}$$

1. Compute the tip error and Jacobian
2. Solve for a proposed joint change
3. Check the **actual cost** with Armijo; shorten the trial if needed
4. Update the joints, rebuild the model, and repeat

Four faces of the same step

K	D	$\mathbf{J}^\#$	what it is
any (J square, invertible)	0	$\mathbf{J}^{-1}$	Deck 03's differential inverse, applied to the remaining displacement (Deck 03's bridge); Gauss-Newton
$w\mathbf{I}, w \rightarrow \infty$	I	$\mathbf{J}^+ (= \mathbf{J}^\top (\mathbf{J}\mathbf{J}^\top)^{-1})$ at full row rank)	w is a positive scalar weight, equal for all task-error components. The limit gives Deck 03's minimum-norm least-squares step; exact when the correction is achievable.
I	$\rho\mathbf{I}$	$(\mathbf{J}^\top \mathbf{J} + \rho\mathbf{I})^{-1} \mathbf{J}^\top$	slide 21's damped step with $\mathbf{J}_r = \mathbf{J}$, since $-\mathbf{J}_r^\top \mathbf{r} = \mathbf{J}^\top (\mathbf{y}^* - \varphi)$; the code and the module
general	general $\succ 0$	$(\mathbf{J}^\top \mathbf{K} \mathbf{J} + \mathbf{D})^{-1} \mathbf{J}^\top \mathbf{K}$	"these joints are cheaper than those"; KOMO weights features this way

Optional, live: the gradient, Gauss-Newton and damped steps from one state on another sum of squares, with the damping adapted step by step

modules/opt-explorer · Trajectory tab, preset "Three steps from one state"

▶ run it live

Worked example: the two-link arm reaches (0.5, 0.2)

start	value
$\mathbf{q}^{(0)}$	$(30^\circ, 45^\circ)$
$\varphi(\mathbf{q}^{(0)})$	$(0.4241, 0.4898)$ m
$f(\mathbf{q}^{(0)})$	0.0449 m ²

$$\mathbf{J}^\#|_{\mathbf{K}=\mathbf{I}, \mathbf{D}=0} = \mathbf{J}^{-1} = \begin{bmatrix} 0.9151 & 3.4151 \\ -4.9975 & -5.7721 \end{bmatrix}, \quad \mathbf{y}^* - \varphi = \begin{bmatrix} 0.0759 \\ -0.2898 \end{bmatrix}$$

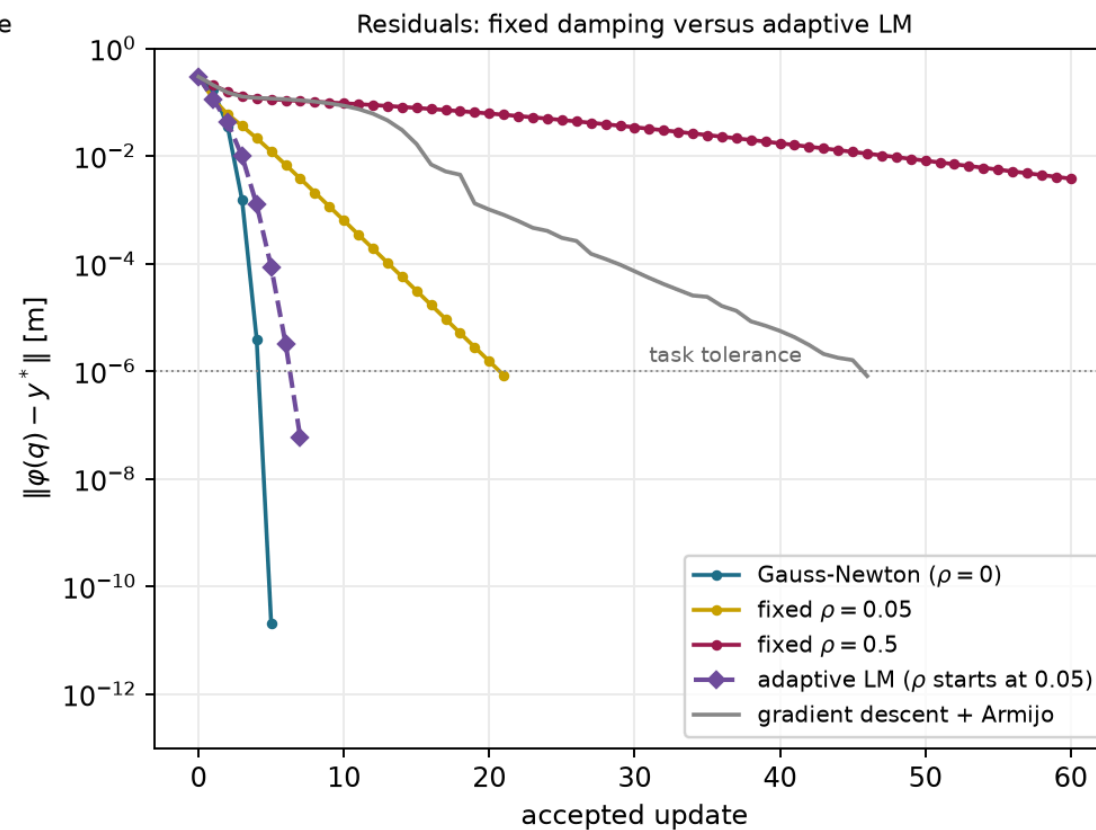
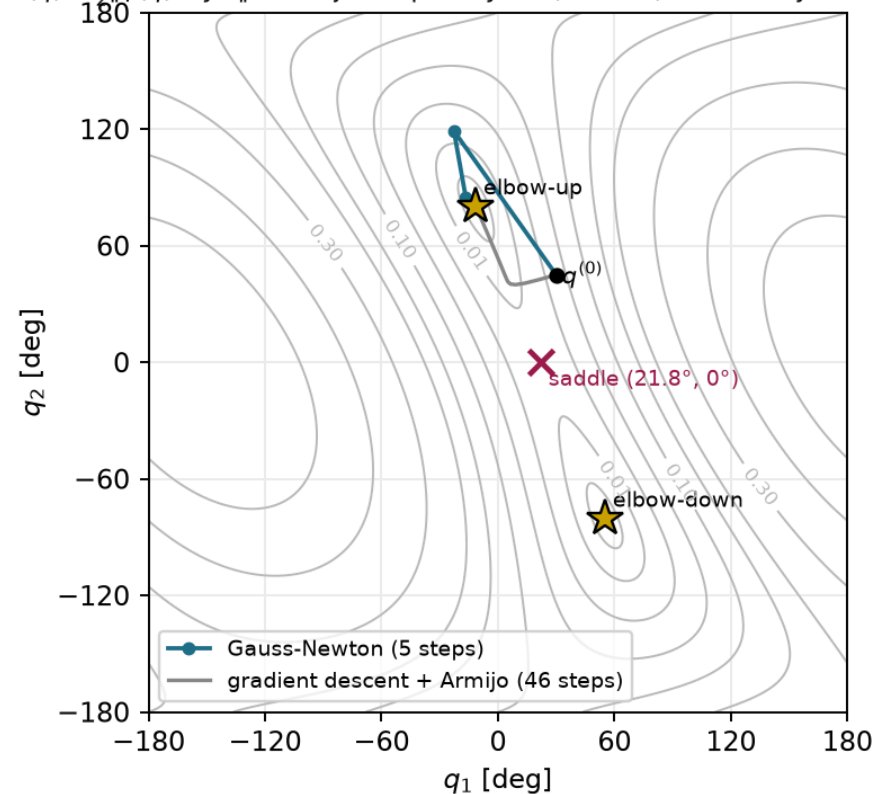
	$\mathbf{D} = 0$	$\mathbf{D} = 0.05 \mathbf{I}$
first step	$(-52.7^\circ, 74.1^\circ)$ $(-0.9201, 1.2931)$ rad	$(-23.8^\circ, 11.5^\circ)$
residual after it	0.17 m	0.115 m
f after it	0.0151 m ²	0.0066 m ²
Armijo, $\alpha = 1$	accepts	accepts
steps	5	21

i	$\mathbf{q}$ (deg)	$\ \varphi - \mathbf{y}^*\ $	$\div$ previous ²
0	(30.0, 45.0)	3.0×10^{-1}	—
1	(-22.7, 119.1)	1.7×10^{-1}	—
2	(-16.5, 84.5)	3.6×10^{-2}	1.2
3	(-11.5, 80.7)	1.5×10^{-3}	1.2
4	(-11.5, 80.4)	4.0×10^{-6}	1.7
5	(-11.5, 80.4)	2.0×10^{-11}	1.3

From step 2 on, each residual is about a constant times the square of the one before (last column). This is **quadratic convergence**.

Iterates and residuals

$f(q) = \frac{1}{2} \|\phi(q) - y^*\|^2$ over joint space, $y^* = (0.5, 0.2)$: two valleys, one saddle

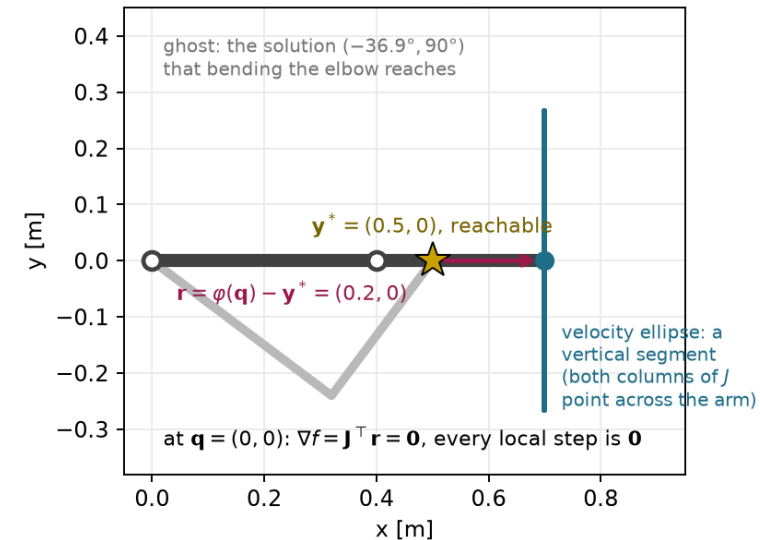


Stop, and say which of four things happened

outcome	test	meaning
solved	$\ \varphi(\mathbf{q}) - \mathbf{y}^*\ < \epsilon$	the target is reached to tolerance
stationary	$\ \mathbf{J}^\top \mathbf{r}\ \leq \epsilon_g \ \mathbf{J}\ _F \ \mathbf{r}\ $ with $\mathbf{r}$ not small ($\mathbf{K} = \mathbf{I}$): the course's relative gradient test	$\nabla f \approx 0$: the least-squares compromise if a minimum, or a bad stationary point
stalled	the line search halves α down to its smallest allowed value, the floor , and Armijo still fails	slow progress, not stationarity (next slide)
limit	the iteration cap, or a step too long to mean anything (diverged)	undecided

None of the four proves that no $\mathbf{q}$ reaches $\mathbf{y}^*$.

Reachable target, zero gradient: stationary, not solved, not unreachable



stretched arm, target on its own line

$$\nabla f = \mathbf{0}$$

every local step is 0

$(-36.9^\circ, 90^\circ)$ reaches the target

When the target is on the boundary, or beyond it

Boundary, $\mathbf{y}^* = (0.7, 0)$: solved, but only linearly

- q_2 halves per step
- the residual is divided by four per step
- the solution is the stretched arm, $\det \mathbf{J} = l_1 l_2 \sin q_2 = 0$: $\mathbf{J}$ moves the tip only sideways, so the linear model sees only part of the gap

Unreachable, $\mathbf{y}^* = (0.8, 0.1)$, plain Gauss-Newton:

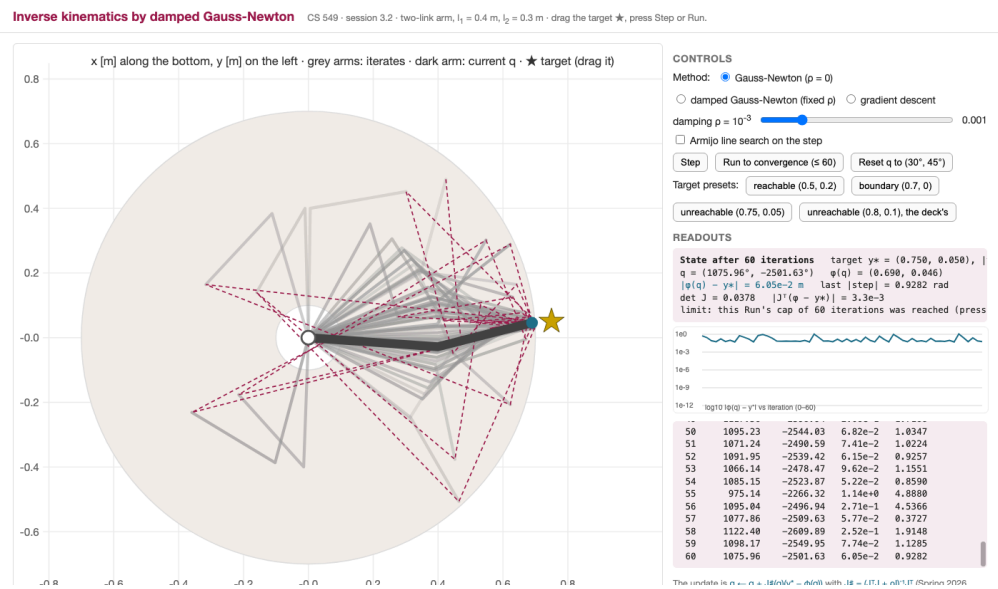
step	what happens
2	reaches $(7.6^\circ, 0.8^\circ)$, almost straight
3	length 49.8 rad
later	wanders to the iteration limit

- closest reachable point: $(0.695, 0.087)$
- residual there: 0.106 m

method	outcome (iteration)	tip to closest point	$\ \nabla f\ $
Gauss-Newton + Armijo, floor 10^{-6}	stalled (7)	9.96 mm	9×10^{-3}
same, floor 10^{-8}	stalled (10)	9.96 mm	9×10^{-3}
same, floor 10^{-12}	stalled (12)	9.96 mm	9×10^{-3}
damped, $\rho = 0.01$	stationary (29)	0.00 mm	5×10^{-10}

The stall is this implementation's: the next halving, 2^{-21} , passes.

Live demo: the Gauss-Newton module

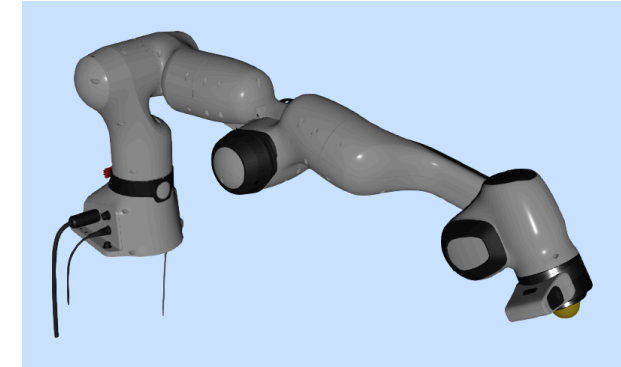


do	read
default: unreachable target (0.75, 0.05), plain Gauss-Newton, Step $\times 7$	elbow 0.5° $\det \mathbf{J} = 0.0011$
Step once more	a step of 35.7 rad
Reset , tick Armijo , Run	on the boundary by step 5 0.4 mm from the closest point readout: stalled , not stationary
untick Armijo, click the label ρ , press $\rightarrow$ three times, Run	$\rho = 5.6 \times 10^{-3}$ within 1 mm by step 4
preset (0.5, 0.2), select Gauss-Newton ($\rho = 0$), Run	quadratic convergence on the log chart, 5 steps
drag the target	anywhere

A real arm: the Panda in Drake

- Position target: $\mathbf{J}$ is 3×7
- $\mathbf{J}^\top \mathbf{J}$ is 7×7 of rank 3: with $\mathbf{D} = 0$, infinitely many solutions;
 $\mathbf{D} = \rho \mathbf{I}$ or $\mathbf{J}^+$ picks one

run	start, target	method	steps	outcome
1	ready, 10 cm sideways and 5 cm down	$\rho = 10^{-3}$	4	solved
2	stretched, 5 cm along the weakest direction	$\rho = 10^{-3}$ $\rho = 10^{-8}$, $\mathbf{J}^+$	18 5	solved $\rho = 10^{-3}$: rate 0.6 (slide 21)
3a	beyond reach	$\rho = 10^{-3}$, full steps	—	limit; steps of 0.9 to 4.2 rad, never settles
3b	beyond reach	Armijo	40	limit, still converging within 1 mm of the stationary reference tip 0.246 m from the target



run 2, solved; target in gold

run 1, step i	residual (m)
0	1.1×10^{-1}
1	1.7×10^{-2}
2	3.7×10^{-4}
3	3.3×10^{-6}
4	5.0×10^{-8}

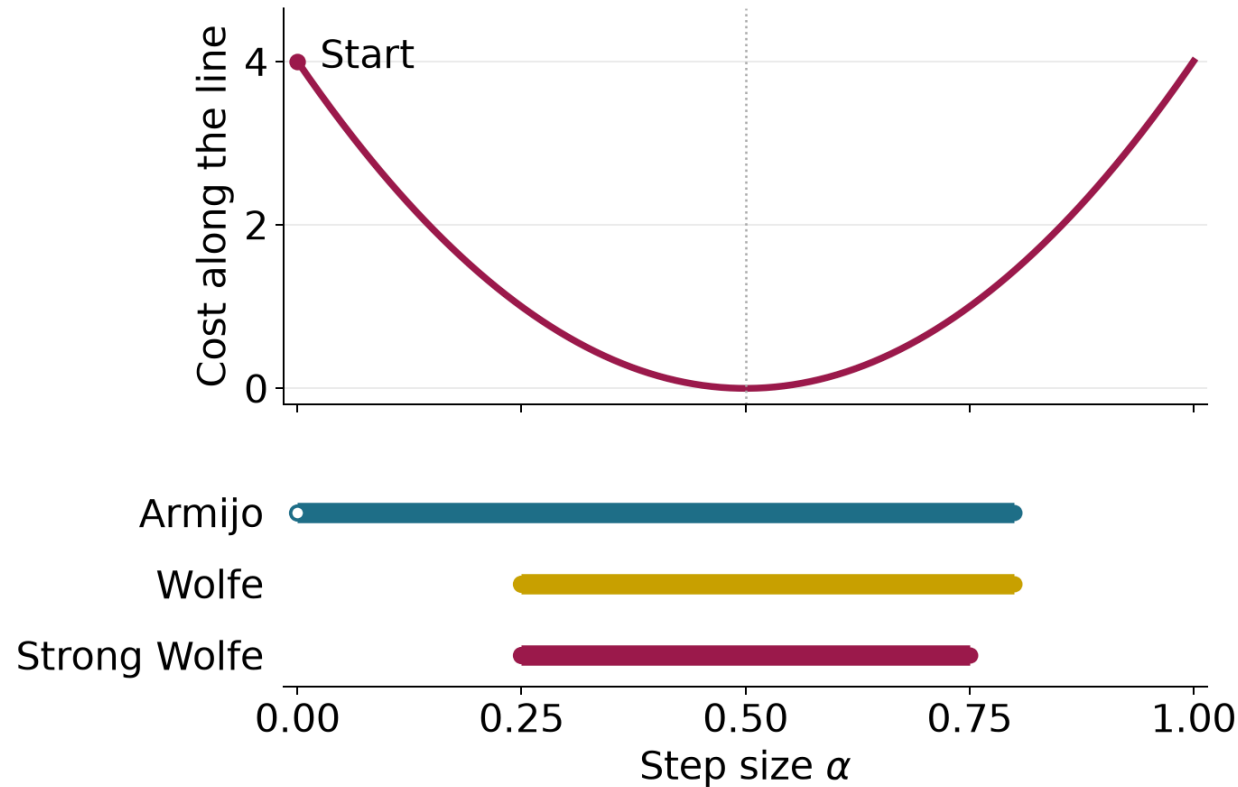
Use previous steps to choose better directions

When computing a full Hessian is expensive:

Method	What it reuses and why
BFGS	Gradient changes reveal curvature; no Hessian calculation
L-BFGS	A short history reduces storage for many variables
Nonlinear conjugate gradient	The previous direction helps guide the new one; little memory

These methods choose a direction. A **line search** still decides how far to move.

A useful step must do more than lower the cost



Armijo: enough cost decrease

Wolfe: also reject steps that remain steeply downhill

Strong Wolfe: also reject slopes that are too steeply uphill

The bands show accepted step sizes along **one fixed direction**.

Summary of 3.2 answers slide 2's three questions

- **How far to trust the step?** Armijo tests the actual fall in f and shortens or rejects a step that misses its threshold. Damping penalizes the proposed step before it is taken. Neither guarantees reaching the target.
- **A $3 \times 7 \mathbf{J}$?** The damped step never inverts $\mathbf{J}$. With $\mathbf{D} = 0$ the step equations have infinitely many solutions, and $\mathbf{D} \succ 0$ picks one (slide 31).
- **What is "closest"?** "Closest" means the globally smallest task error. A small gradient establishes only approximate stationarity, which a saddle or a local minimum also passes (slide 28). For the two-link arm, reach is settled by geometry, $0.1 \text{ m} \leq \|\mathbf{y}^*\| \leq 0.7 \text{ m}$, not by a status word (slides 28 and 29).

What Gauss-Newton does not know

- **Joint limits:** an elbow stop at $|q_2| \leq 60^\circ$

	$\mathbf{q}$	$ q_2 $	result
elbow-up solution	$(-11.5^\circ, 80.4^\circ)$	80.4°	reaches the target, past the stop
elbow-down solution	$(55.1^\circ, -80.4^\circ)$	80.4°	reaches the target, past the stop
clip q_2 after each step	$(-16.4^\circ, 60^\circ)$	60°	stops moving 14.6 cm off

Deck 05 finds the best allowed configuration, still 7.0 cm off.

- **Obstacles:** nothing in $\frac{1}{2} \|\varphi(\mathbf{q}) - \mathbf{y}^*\|^2$ knows the table exists
- **Hard requirements:** "exactly here" versus "as close as possible"

Constraints, multipliers and KOMO come in **Deck 05 (4.1–4.2)**, and its solver runs the same Gauss-Newton step inside.

Failure modes of the day

- **Tuning α by hand** instead of letting Armijo decide per step
- **Scaling:** α has units, rad^2/m^2 on the arm, where it settles near 5. Iterations on the arm:
 - α capped at 1: 312
 - uncapped: 46
- **Trusting the linear model far away:** two cases of the same mistake
 - slide 26's first step $(-52.7^\circ, 74.1^\circ)$: Armijo accepts it, damping shortens it
 - slide 29's 49.8 rad step: Armijo rejects it
- **Mixing up the three expressions:**
 - $-\mathbf{J}_r^\top \mathbf{r}$ is gradient descent
 - $-(\mathbf{J}_r^\top \mathbf{J}_r)^{-1} \mathbf{J}_r^\top \mathbf{r}$ is Gauss-Newton
 - $-\mathbf{J}_r$ alone is a shape error
- **Stopping on one number:** report both norms and the outcome of slide 28; a small gradient with a large residual is a stationary point, the least-squares compromise if it is a minimum, and never a proof that $\mathbf{y}^*$ is unreachable
- **Reading a floor as a theorem:** "stalled" is the implementation's stopping rule; a smaller trial can pass, and the slow progress belongs to the direction

A1 part 4 is now open; R 3; the next deck

- **A1 part 4:** implement $\mathbf{q} \leftarrow \mathbf{q} + \mathbf{J}^\#(\mathbf{y}^* - \varphi(\mathbf{q}))$ on your own two-link arm, position target; report the outcome of slide 28 with both norms; the trajectory part waits for Deck 05
- **R 3:** quiz 1 (weeks 1–2); the teaching assistants' RAI tutorial
- **Deck 05 (4.1–4.2):** constraints and the Karush–Kuhn–Tucker (KKT) conditions from a force balance; inverse kinematics and trajectories as one nonlinear program (KOMO)

Backup slides

The main lecture introduces solver choices and line-search checks at a practical level. These backups retain the update formulas, detailed conditions and algorithms. The frozen Spring 2026 sources (`source-2026/opt-lectures/01, 02, 04`) and the inherited app provide further reference.

Backup · Wolfe conditions: Armijo plus a curvature check

$$(1) f(\mathbf{z} + \alpha \mathbf{d}) \leq f(\mathbf{z}) + c_1 \alpha \nabla f(\mathbf{z})^\top \mathbf{d} \quad (2) \nabla f(\mathbf{z} + \alpha \mathbf{d})^\top \mathbf{d} \geq c_2 \nabla f(\mathbf{z})^\top \mathbf{d}$$

- (1) is Armijo: enough decrease. (2) is **curvature**: the slope along $\mathbf{d}$ must have flattened enough; it rules out lazy tiny steps
- Strong Wolfe replaces (2) by $|\nabla f(\mathbf{z} + \alpha \mathbf{d})^\top \mathbf{d}| \leq c_2 |\nabla f(\mathbf{z})^\top \mathbf{d}|$

typical constant	Newton and quasi-Newton directions	nonlinear conjugate gradients
c_1	10^{-4}	10^{-4}
c_2	0.9	0.1

Backup · Bracket and zoom

```
bracket:   $\alpha \leftarrow \bar{\alpha}$  ; repeat: if Armijo fails or  $f_d(\alpha) \geq f_d(\text{previous})$ : zoom(previous,  $\alpha$ )
        if  $|f'_d(\alpha)| \leq c_2 |f'_d(0)|$ : return  $\alpha$                                 # strong Wolfe met
        if  $f'_d(\alpha) \geq 0$ : zoom( $\alpha$ , previous)
         $\alpha \leftarrow \min(2\alpha, \alpha_{\max})$                                 # grow
zoom(lo, hi): repeat:  $\alpha \leftarrow (lo + hi)/2$ 
        if Armijo fails or  $f_d(\alpha) \geq f_d(lo)$ : hi  $\leftarrow \alpha$ 
        else: if  $|f'_d(\alpha)| \leq c_2 |f'_d(0)|$ : return  $\alpha$ 
               if  $f'_d(\alpha)(hi - lo) \geq 0$ : hi  $\leftarrow lo$ 
               lo  $\leftarrow \alpha$ 
```

$f_d(\alpha) = f(\mathbf{z} + \alpha \mathbf{d})$. Grow first to find an interval that must contain a Wolfe step, then bisect it.

Backup · Steepest descent in a general norm

$$\mathbf{d}_M = \arg \min_{\mathbf{d}} \nabla f(\mathbf{z})^\top \mathbf{d} \quad \text{subject to } \|\mathbf{d}\|_M = \sqrt{\mathbf{d}^\top \mathbf{M} \mathbf{d}} \leq 1 \quad \Rightarrow \quad \mathbf{d}_M \propto -\mathbf{M}^{-1} \nabla f(\mathbf{z})$$

- $\mathbf{M} = \mathbf{I}$: gradient descent
- $\mathbf{M} = \mathbf{H}$: **Newton's direction**, the steepest descent in the metric of the Hessian
- Newton's direction is gradient descent after a change of coordinates that makes the elongated bowl round, which is why conditioning does not slow it
- Any positive definite $\mathbf{M}$ is a **preconditioner**; a diagonal one costs $O(d)$ per step and can remove bad scaling of the coordinates

Backup · Quasi-Newton: BFGS and L-BFGS

Secant condition $\hat{\mathbf{H}} \mathbf{y} = \mathbf{s}$, for the step $\mathbf{s}$ and the gradient change $\mathbf{y}$:

$$\mathbf{s} = \mathbf{z}^{(i+1)} - \mathbf{z}^{(i)}, \quad \mathbf{y} = \nabla f(\mathbf{z}^{(i+1)}) - \nabla f(\mathbf{z}^{(i)})$$

$$\hat{\mathbf{H}}^{(i+1)} = \left(\mathbf{I} - \frac{\mathbf{s} \mathbf{y}^\top}{\mathbf{y}^\top \mathbf{s}} \right) \hat{\mathbf{H}}^{(i)} \left(\mathbf{I} - \frac{\mathbf{y} \mathbf{s}^\top}{\mathbf{y}^\top \mathbf{s}} \right) + \frac{\mathbf{s} \mathbf{s}^\top}{\mathbf{y}^\top \mathbf{s}}, \quad \mathbf{d}^{(i)} = -\hat{\mathbf{H}}^{(i)} \nabla f(\mathbf{z}^{(i)})$$

- $\hat{\mathbf{H}} \approx \mathbf{H}^{-1}$ built from gradients only; stays positive definite if $\mathbf{y}^\top \mathbf{s} > 0$ (Wolfe guarantees it)
- BFGS costs $O(d^2)$ per step;
L-BFGS stores only the last 5 to 20 pairs $(\mathbf{s}, \mathbf{y})$, at $O(d)$ per stored pair

<code>scipy.optimize.minimize</code>	picks
by default	BFGS
with bounds	L-BFGS-B
with constraints	SLSQP

modules/opt-explorer · Trajectory tab, preset "Three steps from one state"

► run it live

Backup · Conjugate gradients

```
g_old ← ∇f(z); d ← -g_old
repeat: α ← argmin_α f(z + α d)           # (near-)exact line search
        z ← z + α d; g_new ← ∇f(z)
        β ← max( g_new⊤(g_new - g_old) / (g_old⊤ g_old), 0 )    # Polak-Ribière
        d ← -g_new + β d; g_old ← g_new
```

- Momentum on the previous direction makes successive directions **conjugate**, $\mathbf{d}^{(i+1)\top} \mathbf{H} \mathbf{d}^{(i)} = 0$, without ever forming $\mathbf{H}$
- On a d -dimensional quadratic: exact minimum in d steps; memory $O(d)$
- Needs an accurate line search; sensitive to non-quadratic behavior