Non-Linear Programming (NLP): Single-Variable, Unconstrained

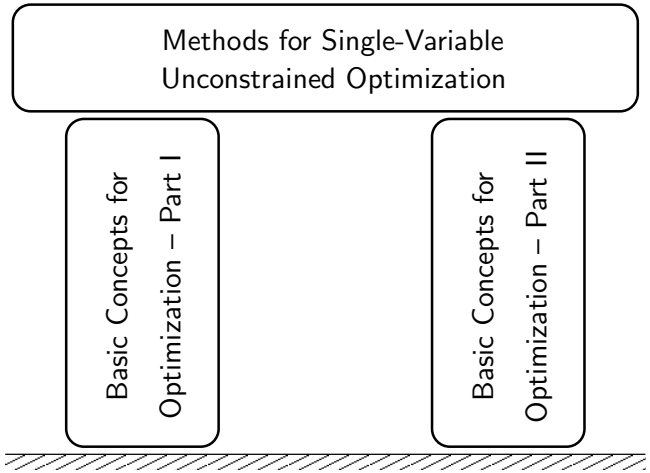
Benoît Chachuat <benoit@mcmaster.ca>

McMaster University
Department of Chemical Engineering

ChE 4G03: Optimization in Chemical Engineering

Solving Single-Variable, Unconstrained NLPs

Prerequisites:



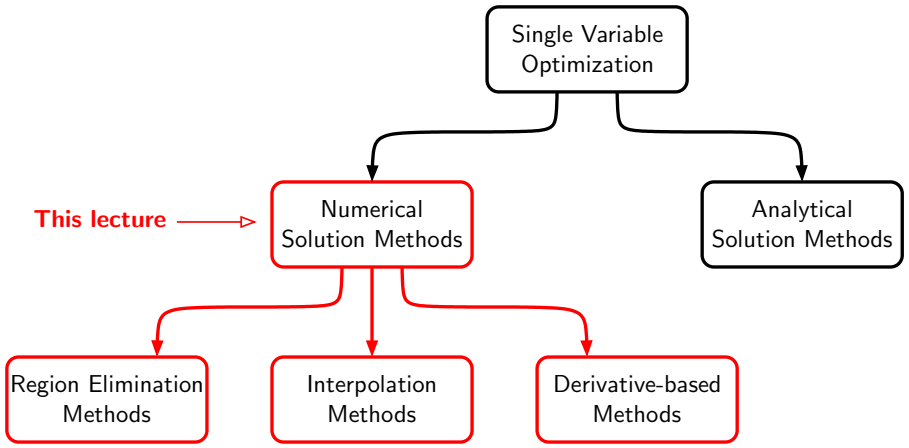
Solving Single-Variable, Unconstrained NLPs

- "Aren't single-variable problems easy?" — Sometimes
- "Won't the methods for multivariable problems work in the single variable case?" — Yes

But,

1. A few important problems are single variables — e.g., nonlinear regression
2. This will give us insight into multivariable solution techniques
3. Single-variable optimization is a subproblem for many nonlinear optimization methods and software! — e.g., linesearch in (un)constrained NLP

Outline

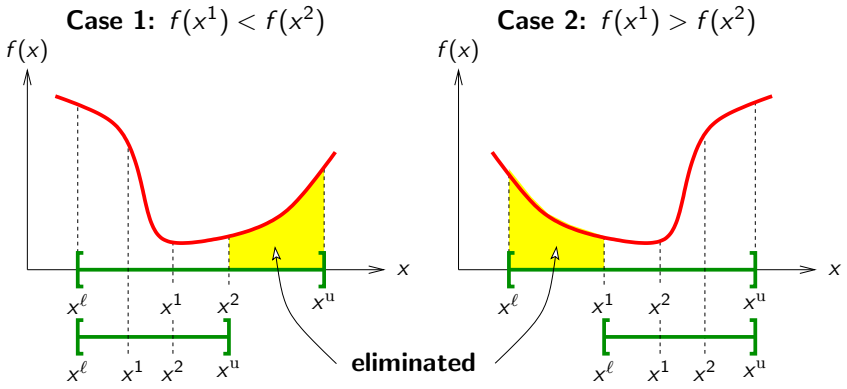


For additional details, see Rardin (1998), Chapter 13.2

Region Elimination Methods (Minimize Case)

Iteratively consider the function value at 4 carefully spaced points:

- Assume a unimodal function
- Leftmost x^ℓ is always a lower bound on the optimal value x^*
- Rightmost x^u is always an upper bound on the optimal value x^*
- Points x^1 and x^2 fall in between



Golden Section Search (Minimize Case)

How do we choose the intermediate points x^1, x^2?

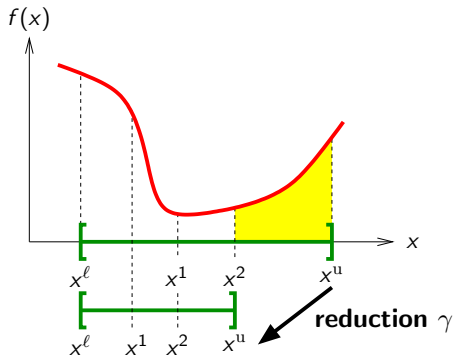
- Golden section search proceeds by keeping both possible intervals $[x^\ell, x^2]$ or $[x^1, x^u]$ equal

Golden Section Search

The points x^1, x^2 are taken as:

$$x^1 \triangleq x^u - \gamma\left[x^u - x^\ell\right]$$
$$x^2 \triangleq x^\ell + \gamma\left[x^u - x^\ell\right]$$

with $\gamma \triangleq \dfrac{1+\sqrt{5}}{2} \approx 0.618$ a fraction know as the **golden ratio**



What is the advantage of choosing this ratio γ?

Golden Section Search Algorithm (Minimize Case)

- **Step 0: Initialization**
  - Choose lower and upper bounds, x^ℓ and x^u, that bracket x^*, as well as stopping tolerance $\epsilon > 0$
  - Set $x^1 \leftarrow x^u - \gamma\left[x^u - x^\ell\right]$ and $x^2 \leftarrow x^\ell + \gamma\left[x^u - x^\ell\right]$
- **Step 1: Stopping**
  - If $x^u - x^\ell < \epsilon$, stop — report $x^* \leftarrow \frac{1}{2}\left[x^u - x^\ell\right]$ as an approximate solution
- **Step 2a: Case $f(x^1) < f(x^2)$ (optimum left)**
  - Narrow the search by eliminating the rightmost part:

    $$x^u \leftarrow x^2, \quad x^2 \leftarrow x^1, \quad x^1 \leftarrow x^u - \gamma\left[x^u - x^\ell\right],$$

    and evaluate $f(x^1)$ — Return to step 1
- **Step 2b: Case $f(x^1) > f(x^2)$ (optimum right)**
  - Narrow the search by eliminating the leftmost part:

    $$x^\ell \leftarrow x^1, \quad x^1 \leftarrow x^2, \quad x^2 \leftarrow x^\ell + \gamma\left[x^u - x^\ell\right],$$

    and evaluate $f(x^2)$ — Return to step 1

Pros and Cons of Golden Section Search

Pros:

- Objective function can be nonsmooth or even discontinuous
- Calculations are straightforward (only function evaluations)

Cons:

- Assumes a unimodal function and requires prior knowledge of an enclosure $x^* \in [x^\ell, x^u]$
- Golden section search is **slow**! Considerable computation may be needed to get an accurate solution

Example: Consider the problem: min $f(x) \triangleq \frac{(x-20)^4}{500} - 2x$

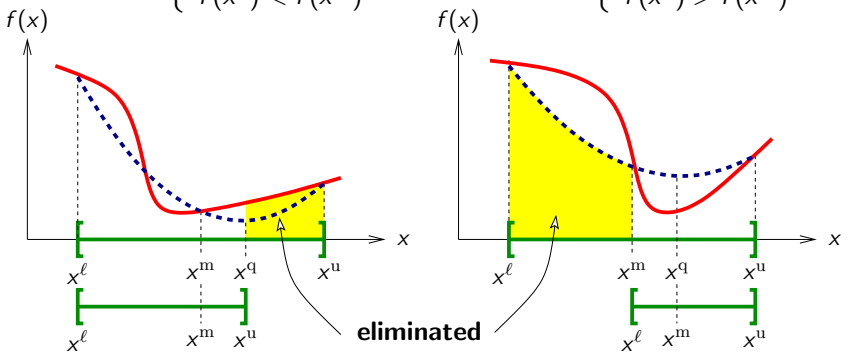
it.	x^ℓ	x^1	x^2	x^u	$f(x^\ell)$	$f(x^1)$	$f(x^2)$	$f(x^u)$	$x^u - x^\ell$
0	0.00	15.28	24.72	40.00	320.00	-29.57	-48.45	240.00	**40.00**
1	15.28	24.72	30.54	40.00	-29.57	-48.45	-36.27	240.00	**24.72**
2	15.28	21.12	24.72	30.56	-29.57	-42.23	-48.45	-36.27	**15.28**
3	21.12	24.72	26.95	30.56	-42.23	-48.45	-49.23	-36.27	**9.44**
4									

Interpolation Methods (Minimize Case)

Improve speed by taking full advantage of a 3-point pattern:

- Assume a unimodal, continuous function
- Fit a smooth curve through the points: $(x^\ell, f(x^\ell))$, $(x^{\mathrm{m}}, f(x^{\mathrm{m}}))$, $(x^{\mathrm{u}}, f(x^{\mathrm{u}}))$; then, optimize the fitted curve

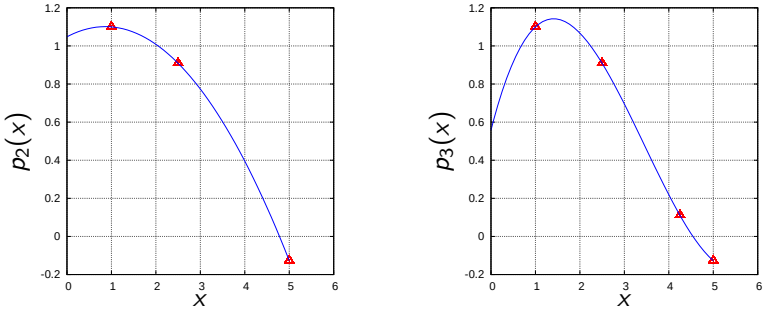
Case 1A: $\begin{cases} x^{\mathrm{q}} > x^{\mathrm{m}}, \text{ and} \\ f(x^{\mathrm{q}}) < f(x^{\mathrm{m}}) \end{cases}$ **Case 1B:** $\begin{cases} x^{\mathrm{q}} > x^{\mathrm{m}}, \text{ and} \\ f(x^{\mathrm{q}}) > f(x^{\mathrm{m}}) \end{cases}$



Math Refresher: Lagrange Polynomials

1. For $N+1$ data points, there is one and only one polynomial of order N, $p_N(x) \triangleq a_0 + a_1 x + \cdots + a_N x^N$, passing through all the points
2. The Lagrange polynomials provide one way of computing this interpolation:

$$p_N(x) \triangleq \sum_{k=0}^{N} p(x_k)\mathcal{L}_k(x), \quad \text{with: } \mathcal{L}_k(x) \triangleq \prod_{\substack{i=1 \\ i\neq k}}^{N} \frac{x - x_i}{x_k - x_i}$$



Quadratic Fit Search (Minimize Case)

How do we fit the curve?

- Quadratic fit search proceeds by fitting a 2nd-order polynomial through $(x^\ell, f(x^\ell))$, $(x^{\mathrm{m}}, f(x^{\mathrm{m}}))$ $(x^{\mathrm{u}}, f(x^{\mathrm{u}}))$:

$$p_2(x) \triangleq f(x^\ell)\frac{(x-x^{\mathrm{m}})(x-x^{\mathrm{u}})}{(x^\ell-x^{\mathrm{m}})(x^\ell-x^{\mathrm{u}})} + f(x^{\mathrm{m}})\frac{(x-x^\ell)(x-x^{\mathrm{u}})}{(x^{\mathrm{m}}-x^\ell)(x^{\mathrm{m}}-x^{\mathrm{u}})} + f(x^{\mathrm{u}})\frac{(x-x^\ell)(x-x^{\mathrm{m}})}{(x^{\mathrm{u}}-x^\ell)(x^{\mathrm{u}}-x^{\mathrm{m}})}$$

Quadratic Fit Search

The unique optimum of the fitted quadratic function occurs at:

$$x^{\mathrm{q}} \triangleq \frac{1}{2}\frac{f(x^\ell)\left[x^{\mathrm{m}2}-x^{\mathrm{u}2}\right] + f(x^{\mathrm{m}})\left[x^{\mathrm{u}2}-x^{\ell 2}\right] + f(x^{\mathrm{u}})\left[x^{\ell 2}-x^{\mathrm{m}2}\right]}{f(x^\ell)\left[x^{\mathrm{m}}-x^{\mathrm{u}}\right] + f(x^{\mathrm{m}})\left[x^{\mathrm{u}}-x^\ell\right] + f(x^{\mathrm{u}})\left[x^\ell-x^{\mathrm{m}}\right]}$$

Quadratic Fit Search Algorithm (Minimize Case)

- **Step 0: Initialization**
  - Choose starting 3-point pattern $(x^\ell, x^{\mathrm{m}}, x^{\mathrm{u}})$, as well as stopping tolerance $\epsilon > 0$
- **Step 1: Stopping**
  - If $x^{\mathrm{u}} - x^\ell < \epsilon$, stop — report $x^* \leftarrow x^{\mathrm{m}}$ as an approximate solution
- **Step 2: Quadratic Fit**
  - Compute quadratic fit optimum x^{q} as

$$x^{\mathrm{q}} \leftarrow \frac{1}{2}\frac{f(x^\ell)\left[x^{\mathrm{m}2}-x^{\mathrm{u}2}\right] + f(x^{\mathrm{m}})\left[x^{\mathrm{u}2}-x^{\ell 2}\right] + f(x^{\mathrm{u}})\left[x^{\ell 2}-x^{\mathrm{m}2}\right]}{f(x^\ell)\left[x^{\mathrm{m}}-x^{\mathrm{u}}\right] + f(x^{\mathrm{m}})\left[x^{\mathrm{u}}-x^\ell\right] + f(x^{\mathrm{u}})\left[x^\ell-x^{\mathrm{m}}\right]}$$

- **Step 3a: Case $x^{\mathrm{q}} < x^{\mathrm{m}}$**
  - If $f(x^{\mathrm{q}}) > f(x^{\mathrm{m}})$, update $x^\ell \leftarrow x^{\mathrm{q}}$
  - Otherwise, update $x^{\mathrm{u}} \leftarrow x^{\mathrm{m}}, \quad x^{\mathrm{m}} \leftarrow x^{\mathrm{q}}$
  - Return to step 1
- **Step 3b: Case $x^{\mathrm{q}} > x^{\mathrm{m}}$**
  - If $f(x^{\mathrm{q}}) > f(x^{\mathrm{m}})$, update $x^{\mathrm{h}} \leftarrow x^{\mathrm{q}}$
  - Otherwise, update $x^\ell \leftarrow x^{\mathrm{m}}, \quad x^{\mathrm{m}} \leftarrow x^{\mathrm{q}}$
  - Return to step 1

Pros and Cons of Quadratic Fit Search

Pros:

- Calculations are also straightforward (only function evaluations)
- Typically faster than Golden section search (but not so much...)

Cons:

- Assumes a unimodal function and requires prior knowledge of an enclosure $x^* \in [x^\ell, x^\mathrm{u}]$
- Objective function has to be smooth

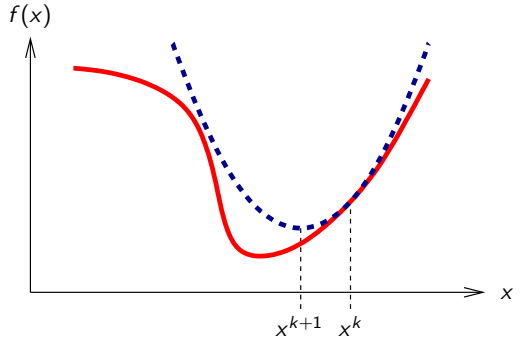
Example: Consider the problem: min $f(x) \triangleq \frac{(x-20)^4}{500} - 2x$

it.	x^ℓ	x^m	x^u	$f(x^\ell)$	$f(x^\mathrm{m})$	$f(x^\mathrm{u})$	$x^\mathrm{u} - x^\ell$	x^q	$f(x^\mathrm{q})$
0	0.00	32.00	40.00	320.00	-22.53	240.00	**40.00**	20.92	-41.84
1	0.00	20.92	32.00	320.00	-41.84	-22.53	**32.00**	25.00	-48.75
2	20.92	25.00	32.00	-41.84	-48.75	-22.53	**11.80**	30.00	-40.03
3	20.92	25.00	30.00	-41.84	-48.75	-40.03	**9.08**		
4									

Derivative-Based Methods

Improve convergence speed by using first- and second-order derivatives taking full advantage of a 3-point pattern:

- Assume a smooth function f
- Consider the quadratic approximation of f passing through the current iterate x^k
- Optimize this approximation to determine the next iterate, x^{k+1}



Newton's Method

How do we get a quadratic approximation at x^k?

- Use Taylor series approximation:

$$f(x) \approx f(x^k) + f'(x^k)\left[x - x^k\right] + \frac{1}{2}f''(x^k)\left[x - x^k\right]^2$$

- Calculate the optimum, x^{k+1}, of the quadratic approximation:

$$f'(x^{k+1}) = 0 = f'(x^k) + f''(x^k)\left[x^{k+1} - x^k\right]$$

Basic Newton's Search

The basic Newton's method proceeds iteratively as

$$x^{k+1} \triangleq x^k - \frac{f'(x^k)}{f''(x^k)}$$

The term $d^{k+1} \triangleq -\frac{f'(x^k)}{f''(x^k)}$ is called the Newton step.

Basic Newton Algorithm

- **Step 0: Initialization**
  - Choose an initial guess x^0, as well as stopping tolerance $\epsilon > 0$
  - Set $k \leftarrow 0$
- **Step 1: Derivatives**
  - Compute first- and second-order derivatives $f'(x^k)$, $f''(x^k)$
- **Step 2: Stopping**
  - If $\left|f'(x^k)\right| < \epsilon$, stop — report $x^* \leftarrow x^k$ as an approximate solution
- **Step 3: Newton Step**
  - Compute Newton step $d^{k+1} \leftarrow -\frac{f'(x^k)}{f''(x^k)}$
  - Update iterate $x^{k+1} \leftarrow x^k + d^{k+1}$
  - Increment $k \leftarrow k+1$ and return to step 1

Variant: Quasi-Newton Algorithm

Idea: Approximate second-order derivatives using finite differences,

$$f''(x^k) \approx \frac{f'(x^k) - f'(x^{k-1})}{x^k - x^{k-1}}$$

- **Step 0: Initialization**
  - Choose initial points x^{-1} and x^0, as well as stopping tolerance $\epsilon > 0$
  - Set $k \leftarrow 0$
- **Step 1: Derivatives**
  - Compute first-order derivative $f'(x^k)$ and approximate inverse of second-order derivatives $B(x^k) \stackrel{\Delta}{=} \frac{x^k - x^{k-1}}{f'(x^k) - f'(x^{k-1})}$
- **Step 2: Stopping**
  - If $\left|f'(x^k)\right| < \epsilon$, stop — report $x^* \leftarrow x^k$ as an approximate solution
- **Step 3: Newton Step**
  - Compute Newton step $d^{k+1} \leftarrow -f'(x^k)B^k$
  - Update iterate $x^{k+1} \leftarrow x^k + d^{k+1}$
  - Increment $k \leftarrow k + 1$ and return to step 1

Pros and Cons of (Quasi-)Newton Search

Pros:

- Very fast convergence close to the optimal solution (quadratic convergence rate)
- No need to bound the optimum within a range

Cons:

- Requires a "good" initial guess, otherwise typically diverges
- No distinction between (local) **minima** and **maxima**!
- Objective function has to be smooth and first-order derivatives must be available (possibly second-order derivatives too)

Example: Consider the problem: min $f(x) \stackrel{\Delta}{=} \frac{(x-20)^4}{500} - 2x$

k	x^k	$f(x^k)$	$f'(x^k)$	$f''(x^k)$	$\lvert f'(x^k)\rvert$	d^{k+1}
0	30.00	-40.00	6.000	2.40	**6.000**	-2.50
1	27.50	-48.67	1.37	1.35	**1.375**	-1.02
2	26.48	-49.43	0.18	1.01	**0.178**	
3						