AI505

Optimization

Local Descent

Marco Chiarandini

Department of Mathematics & Computer Science
University of Southern Denmark

Outline

Preface

For multivariate functions, we have argued that:

- derivatives can have exponential growth in the resulting analytical expression
- calculating zeros might be challenging

Hence, minimizing by solving $\nabla f(\boldsymbol{x}) = 0$ may be computationally demanding.

Descent Direction Iteration

Descent Direction Methods use a local model to incrementally improve design point until some convergence criteria is met

1. Check termination conditions at $\boldsymbol{x}_k$; if not met, continue.
2. Decide **descent direction** $\boldsymbol{d}_k$ using local information
3. Decide **step size** (= magnitude of the overall step = α_k, since commonly $||\boldsymbol{d}_k||_2 = 1$)
4. Compute next design point $\boldsymbol{x}_{k+1}$

$$\boldsymbol{x}_{k+1} \leftarrow \boldsymbol{x}_k + \alpha_k \boldsymbol{d}_k$$

Line Search for Step Size

Assuming we have the search direction:

- Used to compute α
- Using the techniques discussed from previous classes, solve:

$$\text{minimize}_{\alpha} f(\boldsymbol{x} + \alpha \boldsymbol{d})$$

- Often this is computed approximately to reduce cost

Line Search: Alternatives

Step size:

- Fixed α called **learning rate** (commonly $||\boldsymbol{d}_k||_2 = 1$ not imposed)
- **Decaying step factor**

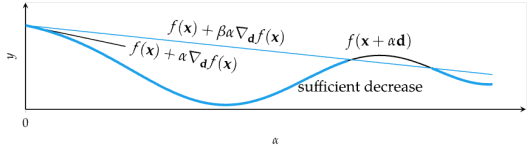
$$\alpha_k = \alpha_1 \gamma^{k-1} \qquad \text{for } \gamma \in [0, 1]$$

Decaying step factor is often required in convergence proofs

Approximate Line Search

- If function calls are expensive, rather than finding the minimum along a search direction, find a point of sufficient decrease

$$f(\boldsymbol{x}_{k+1}) \leq f(\boldsymbol{x}_k) + \beta\alpha\nabla_{\boldsymbol{d}_k} f(\boldsymbol{x}_k)$$

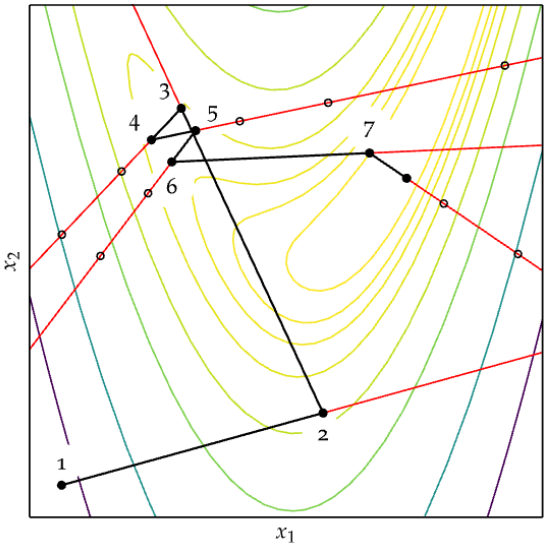


- $\beta \in [0, 1]$, usually $\beta = 1 \times 10^{-4}$

- Backtracking line search starts with a large step and then backs off

```
def backtracking_line_search(f, grad, x, d, alpha_0=1, p=0.5, beta=1e-4):
    y, g, alpha = f(x), grad(x), alpha_0
    while ( f(x + alpha * d) > y + beta * alpha * np.dot(g, d) ) :
        alpha *= p
    return alpha
```

Approximate Line Search: Example



Approximate Line Search

Building on backtracking line search are the Wolfe Conditions each sufficient to guarantee convergence to a local minimum.

1. First Wolfe Condition: Sufficient Decrease

$$f(\boldsymbol{x}_{k+1}) \leq f(\boldsymbol{x}_k) + \beta\alpha\nabla_{\boldsymbol{d}_k} f(\boldsymbol{x}_k)$$

2. Second Wolfe Condition: Curvature Condition

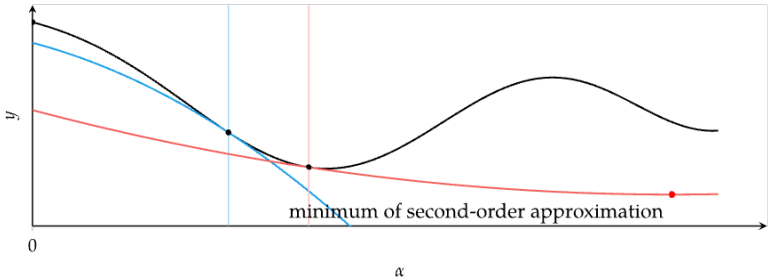
$$\nabla_{\boldsymbol{d}_k} f(\boldsymbol{x}_{k+1}) \geq \sigma\nabla_{\boldsymbol{d}_k} f(\boldsymbol{x}_k)$$

$\beta < \sigma < 1$ with
- $\sigma = 0.1$ with conjugate gradient method
- $\sigma = 0.9$ with Newton method

Approximate Line Search

The curvature condition ensures the second-order function approximations have positive curvature

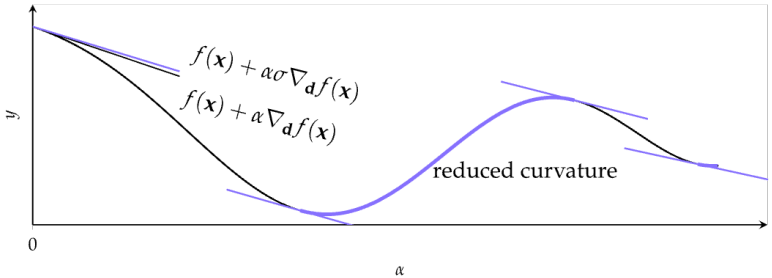
$$\nabla_{\boldsymbol{d}_k} f(\boldsymbol{x}_{k+1}) \geq \sigma \nabla_{\boldsymbol{d}_k} f(\boldsymbol{x}_k)$$



Approximate Line Search

Regions satisfying the curvature condition

$$\nabla_{\boldsymbol{d}_k} f(\boldsymbol{x}_{k+1}) \geq \sigma \nabla_{\boldsymbol{d}_k} f(\boldsymbol{x}_k)$$



Approximate Line Search: Example

Consider approximate line search on $f(x_1, x_2) = x_1^2 + x_1 x_2 + x_2^2$
from $\boldsymbol{x} = [1, 2]$ in the direction $\boldsymbol{d} = [-1, -1]$, gradient at $\boldsymbol{x}$ is $\boldsymbol{g} = [4, 5]$
using a maximum step size of 10, a reduction factor of 0.5,
first Wolfe condition parameter $\beta = 1 \times 10^{-4}$, second Wolfe condition parameter $\sigma = 0.9$.

first Wolfe condition ($f(\boldsymbol{x} + \alpha\boldsymbol{d}) \leq f(\boldsymbol{x}) + \beta\alpha(\boldsymbol{g}^T \cdot \boldsymbol{d})$):

$$\begin{aligned}
\alpha = 10: &\quad f([1,2] + 10 \cdot [-1,-1]) \leq 7 + 1 \times 10^{-4} 10 [4,5]^T [-1,-1] \implies 217 \not\leq 6.991 \\
\alpha = 10 \cdot 0.5 = 5: &\quad f([1,2] + 5 \cdot [-1,-1]) \leq 7 + 1 \times 10^{-4} 5 [4,5]^T [-1,-1] \implies 37 \not\leq 6.996 \\
\alpha = 2.5: &\quad f([1,2] + 2.5 \cdot [-1,-1]) \leq 7 + 1 \times 10^{-4} 2.5 [4,5]^T [-1,-1] \implies 3.25 \leq 6.998
\end{aligned}$$

The candidate design point $\boldsymbol{x}' = \boldsymbol{x} + \alpha\boldsymbol{d} = [-1.5, -.0.5]$ is checked against the second Wolfe condition $\nabla_{\boldsymbol{d}} f(\boldsymbol{x}') \geq \sigma \nabla_{\boldsymbol{d}} f(\boldsymbol{x})$:

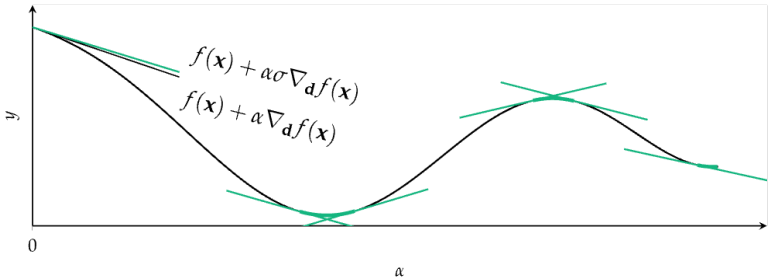
$$[-3.5, -2.5] \cdot [-1, -1] \geq \sigma [4, 5] \cdot [-1, -1] \implies 6 \geq -8.1$$

Approximate line search terminates with $\boldsymbol{x} = [-1.5, -0.5]$.

Approximate Line Search

Regions where the strong curvature condition is satisfied

$$|\nabla_{\boldsymbol{d}_k} f(\boldsymbol{x}_{k+1})| \leq -\sigma \nabla_{\boldsymbol{d}_k} f(\boldsymbol{x}_k)$$



Approximate Line Search

- The sufficient decrease condition with the strong curvature condition form the strong Wolfe conditions.

- Satisfying the strong Wolfe conditions requires a more complicated algorithm

  Strong backtracking line search:

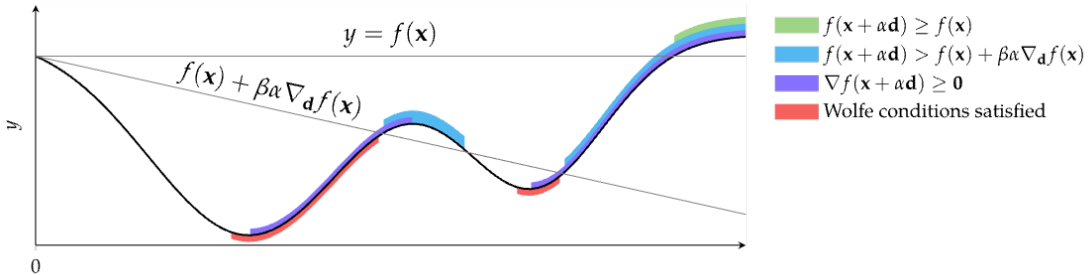
  1. Bracketing Phase: tests successively larger step sizes to bracket an interval $[\alpha_{k-1}, \alpha_k]$ guaranteed to contain step lengths satisfying the Wolfe conditions.
  2. Zoom Phase: shrink the interval using bisection to find point satisfying the **strong** Wolfe conditions

Approximate Line Search

1. Bracketing Phase

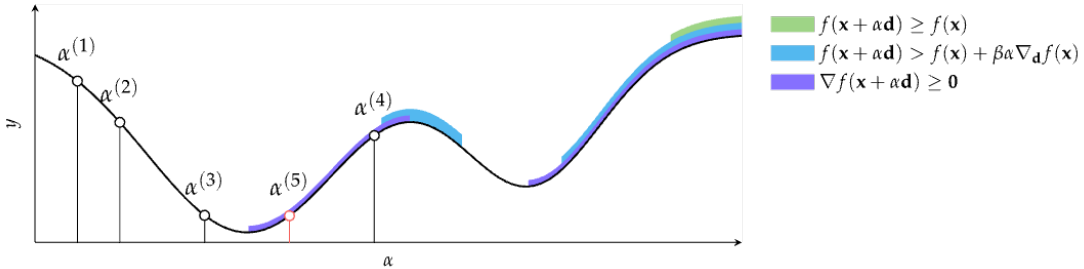
An interval guaranteed to contain step lengths satisfying the Wolfe conditions is found when one of the following conditions hold:

$$f(\boldsymbol{x} + \alpha \boldsymbol{d}) \geq f(\boldsymbol{x})$$
$$f(\boldsymbol{x} + \alpha \boldsymbol{d}) > f(\boldsymbol{x}) + \beta\alpha\nabla \boldsymbol{d} f(\boldsymbol{x})$$
$$\nabla f(\boldsymbol{x} + \alpha \boldsymbol{d}) \geq 0$$



Approximate Line Search

1. Braketing Phase + zoom phase (α_5)



Trust Region Methods

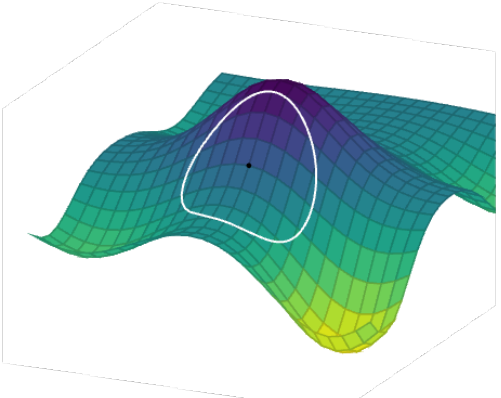
- Descent methods can place too much trust in their first and second order information
- A **trust region** is the local area of the design space where the local model is believed to be reliable.
- Trust region methods, or restricted step methods, limit the step size to ensure local approximation error is minimized
- If the improvement matches the predicted value, the trust region is expanded; otherwise it is contracted

Trust Region Methods

- $\boldsymbol{x}'$ is new design point
- $\hat{f}(\boldsymbol{x}')$ is local function approximation, eg, second-order Taylor approximation
- δ is trust region radius

$$\begin{aligned} &\text{minimize}_{\boldsymbol{x}'}\ \hat{f}(\boldsymbol{x}') \\ &\text{subject to} \quad ||\boldsymbol{x} - \boldsymbol{x}'|| \leq \delta \end{aligned}$$

Constrained optimization problem.
It can be solved efficiently if $\hat{f}$ quadratic



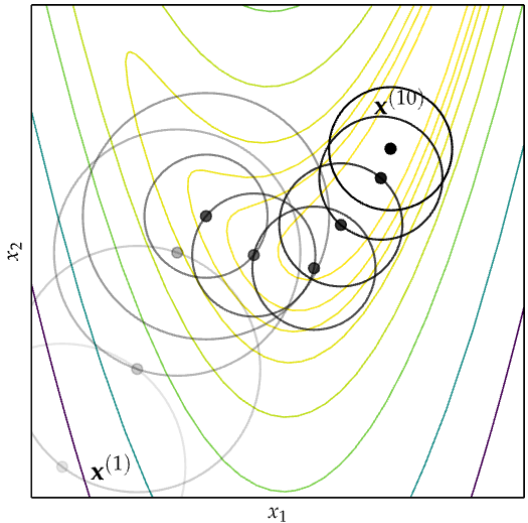
Trust Region Methods

δ can be expanded or contracted based on performance

$$\eta = \frac{\text{actual improvement}}{\text{predicted improvement}} = \frac{f(\boldsymbol{x}) - f(\boldsymbol{x}')}{f(\boldsymbol{x}) - \hat{f}(\boldsymbol{x}')}$$

If $\eta < \eta_1$ contract
if $\eta > \eta_2$ expand

Trust Region Methods: Example



Trust regions can be also non circular.

Trust Region Methods

Termination Conditions (commonly used together):

- Maximum Iterations: $k > k_{\max}$
- Aboslute Improvement: $f(\boldsymbol{x}_k) - f(\boldsymbol{x}_{k+1}) < \epsilon_a$
- Relative Improvement: $f(\boldsymbol{x}_k) - f(\boldsymbol{x}_{k+1}) < \epsilon_r |f(\boldsymbol{x}_k)|$
- Gradient Magnitude: $||\nabla f(\boldsymbol{x}_{k+1})|| < \epsilon_g$

Then random restart.

Summary

- Descent direction methods incrementally descend toward a local optimum.
- Univariate optimization can be applied during line search.
- Approximate line search can be used to identify appropriate descent step sizes.
- Trust region methods constrain the step to lie within a local region that expands or contracts based on predictive accuracy.
- Termination conditions for descent methods can be based on criteria such as the change in the objective function value or magnitude of the gradient.