CME 216, ME 343 - Winter 2021

Eric Darve, ICME

The optimizers we have seen until now belong to the class of first-order optimizers.

This is because to a large extent they only rely on the gradient of the loss function.

For physics-informed learning, we are often interested in converging the loss function to very small values.

This is because high-accuracy is required for the solution and the loss function is often ill-conditioned.

For this reason, optimizers that have improved convergence properties are desirable.

Second order optimizers make use of information about the Hessian to improve convergence.

We will cover three methods:

- Trust Region
- BFGS; Broyden-Fletcher-Goldfarb-Shanno
- L-BFGS or limited-memory BFGS

This is a large and complex topic.

We will only cover the main ideas to give you a general sense of how these methods work.

Trust-region methods start from a quadratic approximation of the loss function.

To follow the standard notations in the field, we use

$$g_k = \nabla l(x_k)$$

$$B_k = \nabla^2 l(x_k)$$

$$l(x_k + p) \approx l(x_k) + g_k^T p + \frac{1}{2} p^T B_k p$$

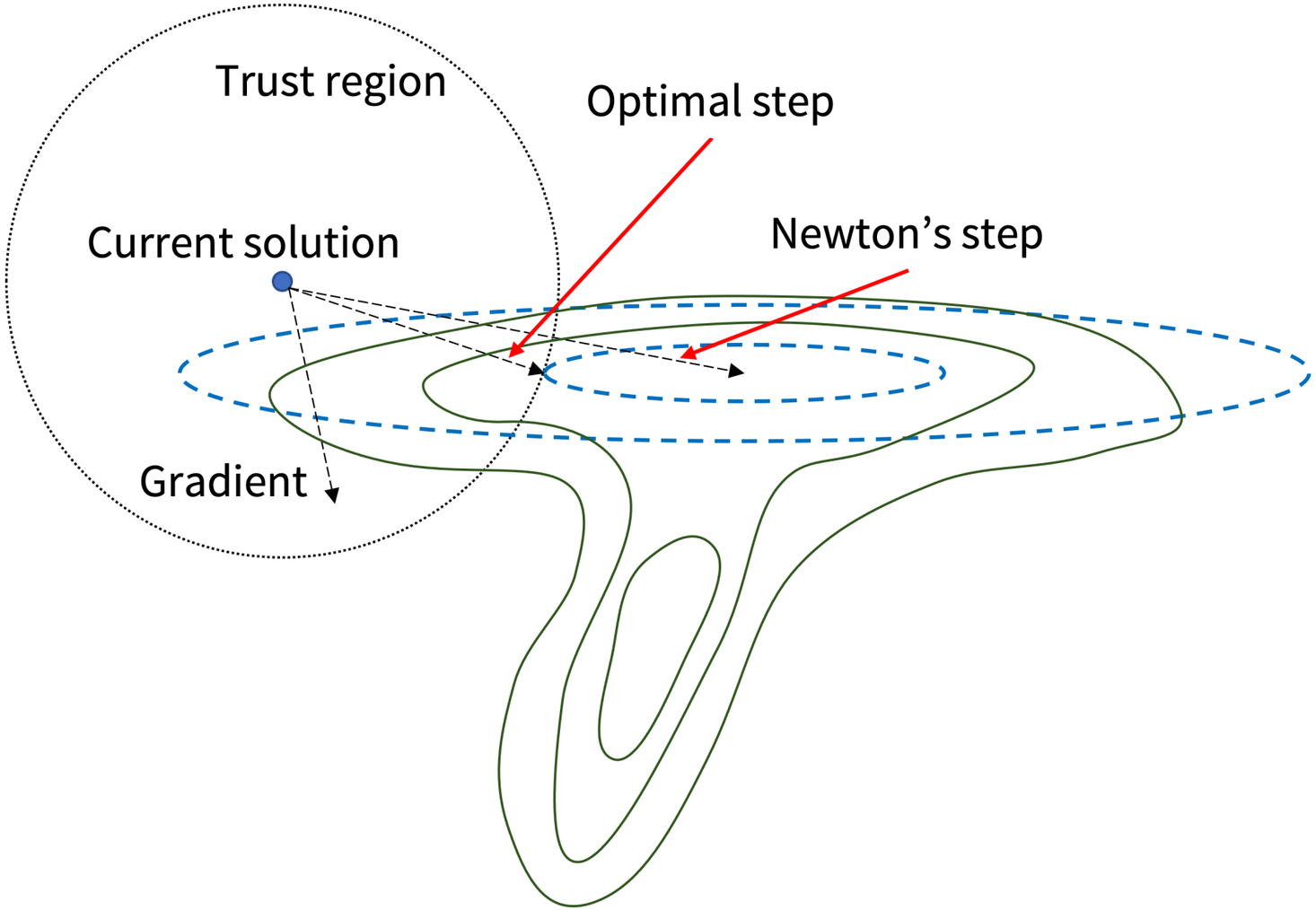
The minimum of that approximation is given by Newton's point:

$$p = -B_k^{-1} g_k$$

However, p in some cases may be too large.

If p is too large, the quadratic approximation may no longer be a good approximation of l.

So we need to limit the size of the step p we take.


Trust region
Optimal step
Current solution
Newton’s step
Gradient

Mathematically the problem we want to solve is:

$$p^* = \mathbf{argmin}_p \; g_k^T p + \frac{1}{2} p^T B_k p$$

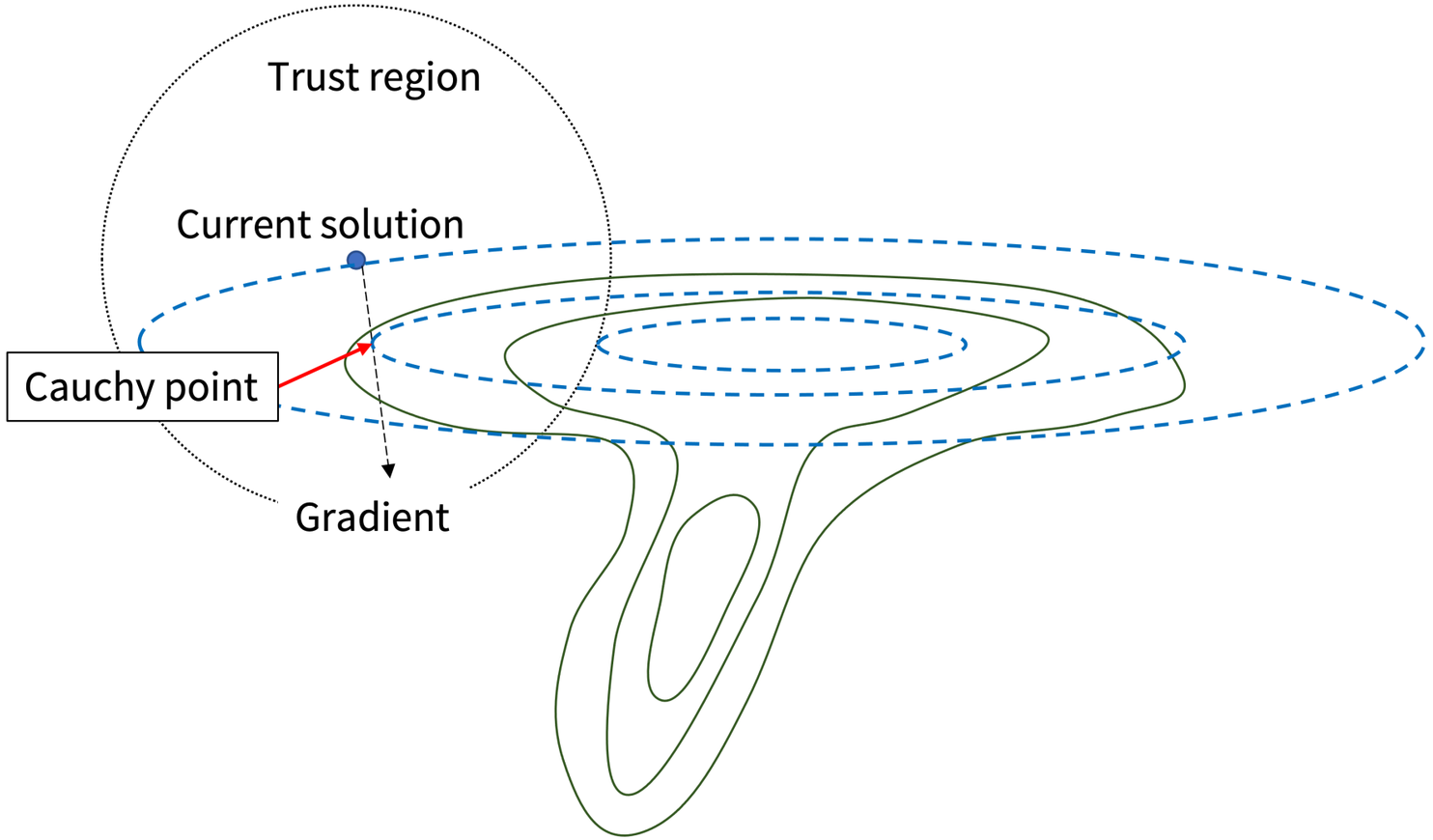
subject to

$$\|p^*\| \leq \Delta$$

The simplest method to find an approximate solution is the Cauchy point.

This is simply a point in the direction of the gradient that minimizes

$$p^* = \mathbf{argmin}_p \; g_k^T p + \frac{1}{2} p^T B_k p$$


Trust region
Current solution
Cauchy point
Gradient

Although we did use the 2nd order approximation to estimate the Cauchy point, we are still following the gradient vector.

Is it possible to find a better approximation of

$$p^* = \mathbf{argmin}_p \; g_k^T p + \frac{1}{2} p^T B_k p$$

subject to

$$\|p^*\| \leq \Delta$$

If $\|p^*\| < \Delta$, then: $p_k = -B_k^{-1} g_k$

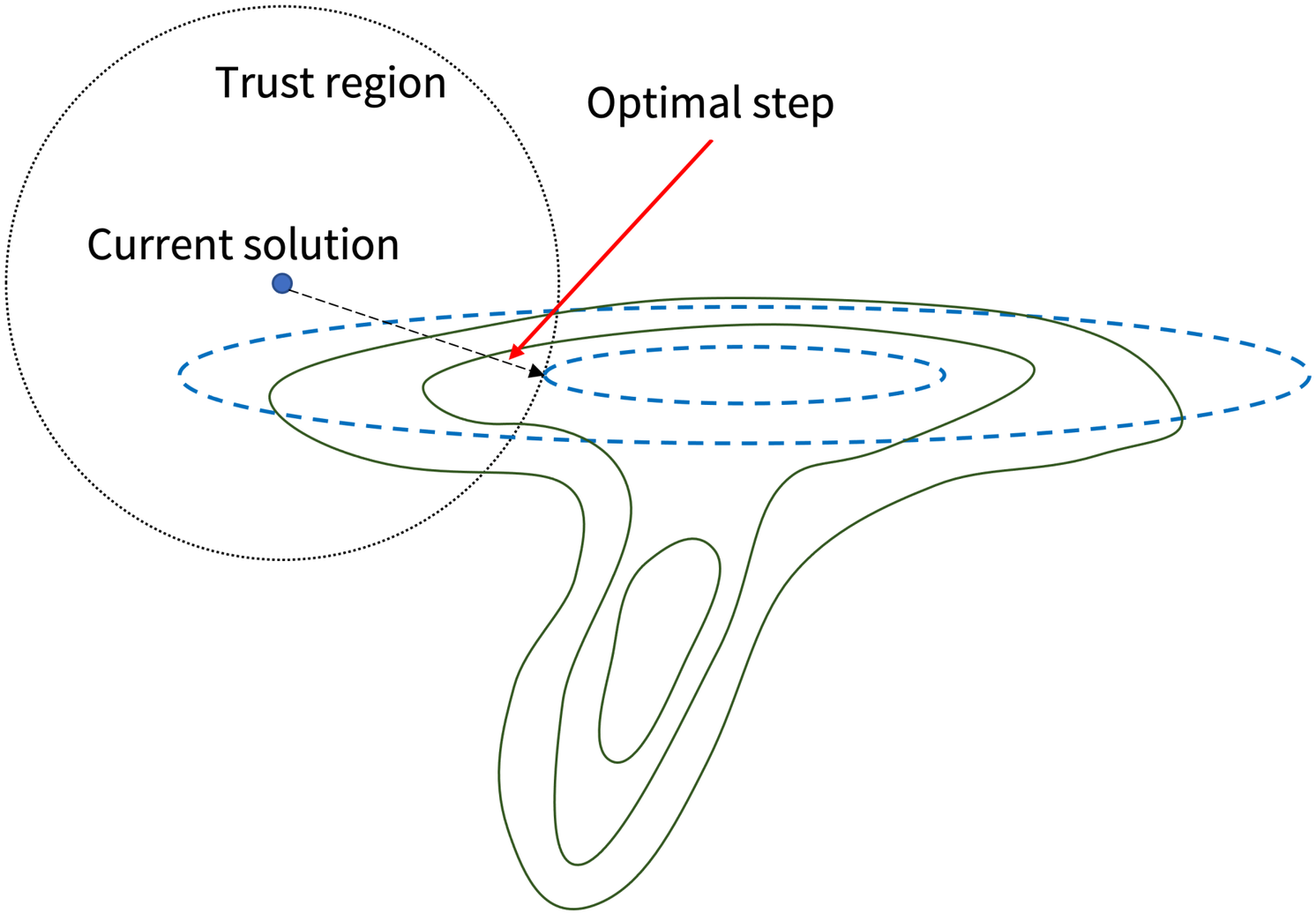
Assume now that $\|p^*\| = \Delta$.

Using the method of Lagrange multipliers, we find that at the optimum, the gradient of the approximate loss function:

$$g_k + B_k p^*$$

is parallel to the gradient of the constraint:

$$p^*$$


Trust region
Optimal step
Current solution

So there is a λ such that:

$$g_k + B_k p^* = \lambda p^*$$

$$p^* = -(B_k - \lambda I)^{-1} g_k$$

Use the eigendecomposition of B_k: $B_k = Q\Lambda Q^T$.

$$p^* = -(B_k - \lambda I)^{-1} g_k$$

$$p^* = -\sum_j \frac{q_j^T g_k}{\lambda_j - \lambda} q_j$$

We look for λ such that $\|p^*\|_2^2 = \Delta^2$.

We get:

$$\sum_j \left(\frac{q_j^T g_k}{\lambda_j - \lambda} \right)^2 = \Delta^2$$

Although complicated, this equation can be solved.

The solution process is facilitated by the fact that λ is simply a real number.

scipy implementations

- dogleg
- trust-exact
- trust-ncg
- trust-krylov

dogleg

Dog-leg method.

Interpolation between Cauchy-point and Newton's point

trust-exact

Exact solution of the trust region subproblem

trust-ncg

Conjugate Gradient iterative solution

It does not require the eigendecomposition of the Hessian.

Only matrix-vector products with B_k are required.

trust-krylov

Similar to `trust-ncg`. Only matrix-vector products with B_k are required.

The Lanczos method is used instead of the Conjugate Gradient.

The method is slightly more expensive but is more accurate and may converge faster.

For more information see Numerical Optimization, by Nocedal and Wright, Springer, 2nd edition