LBFGS

John Langford, Large Scale Machine Learning Class, February 5

(post presentation version)

We are still doing Linear Learning

Features: a vector $x \in \mathbb{R}^n$

Label: $y \in \mathbb{R}$

Goal: Learn $w \in \mathbb{R}^n$ such that $\hat{y}_w(x) = \sum_i w_i x_i$ is close to y.

But, this time in a batch fashion

Initialize w

Repeatedly:

1. Let $\hat{y}_w(x) = \sum_i w_i x_i$
2. Let $g_i = \sum_{(x,y)} \frac{\partial L(\hat{y}_w(x), y)}{\partial w_i}$
3. Compute update direction $d(g)$
4. Update weights $w_i \leftarrow w_i + d_i(g)$

The BFGS Update

$d(g) = Dg$ for some Direction matrix D
What is D?

The BFGS Update

$d(g) = Dg$ for some Direction matrix D

What is D?

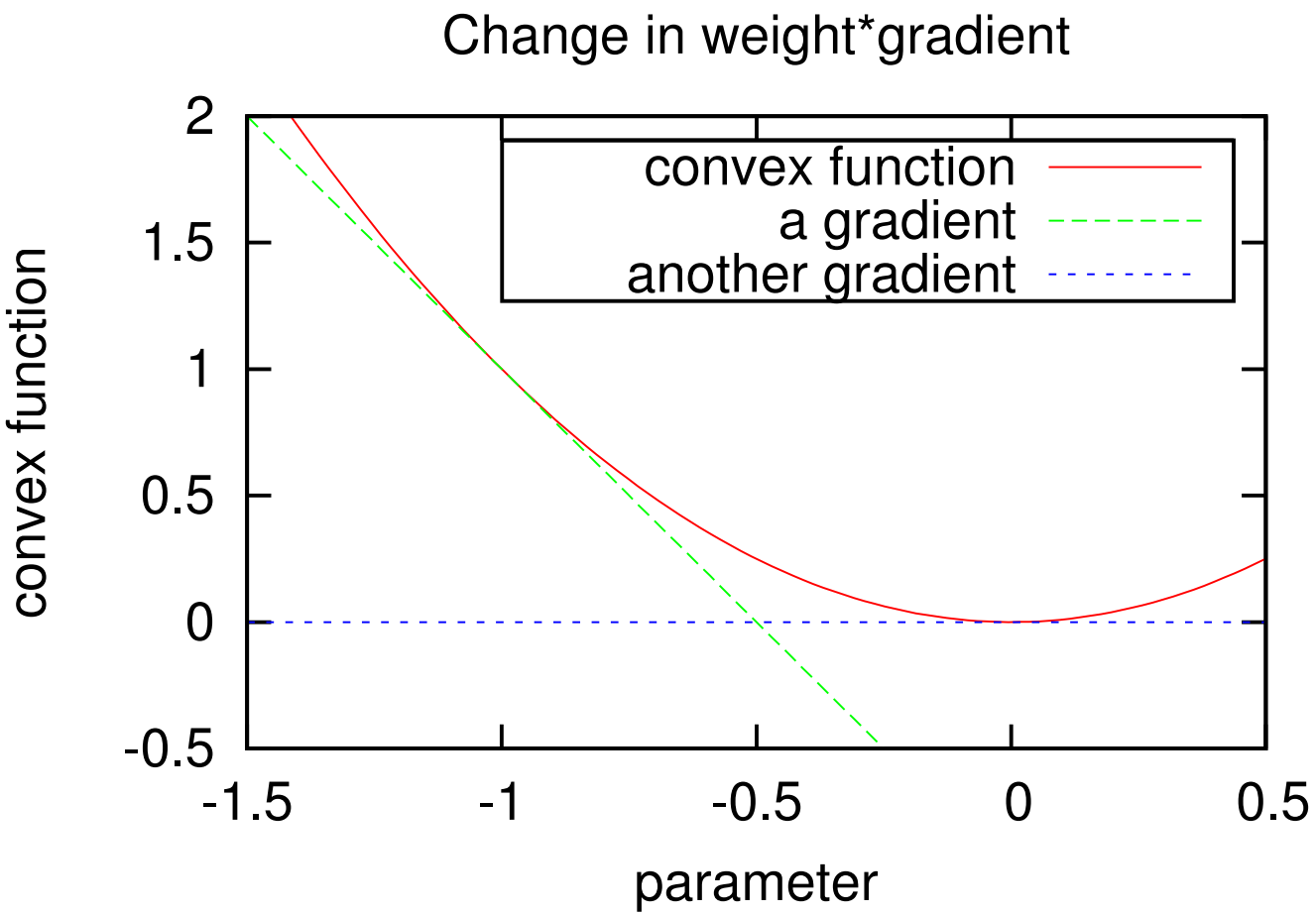
D is defined purely in terms of two empirical observations:

$g' = g_{\text{new}} - g_{\text{prev}}$

$w' = w_{\text{new}} - w_{\text{prev}}$

Assertion 1

$\sum_i g'_i w'_i = g'^{\top} w'$ should be positive for convex functions.



Assertion 2

$T_{kj} = \frac{g'_k w'_j}{\sum_i g'_i w'_i} = \frac{g' w'^{\top}}{g'^{\top} w'}$ Transforms direction g' to direction w' and vice versa.

Assertion 2

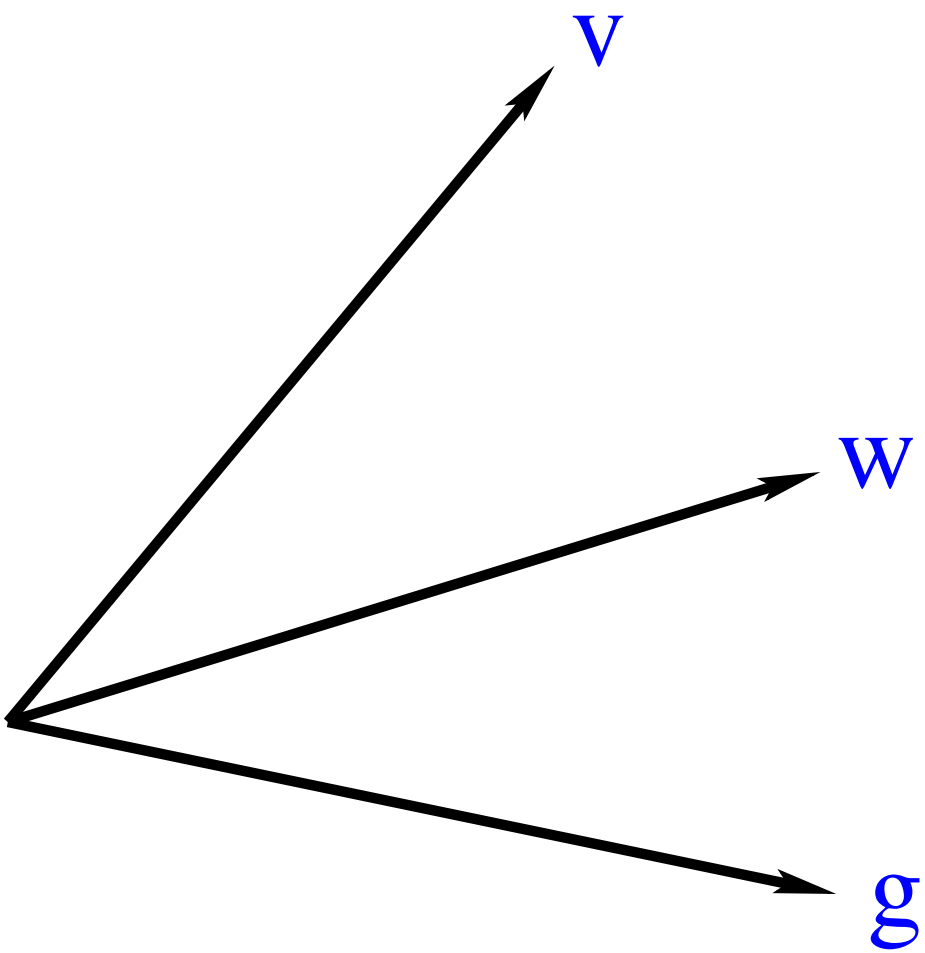
$T_{kj} = \frac{g'_k w'_j}{\sum_i g'_i w'_i} = \frac{g' w'^{\top}}{g'^{\top} w'}$ Transforms direction g' to direction w' and vice versa.

A matrix is a linear function which transforms one vector into another.

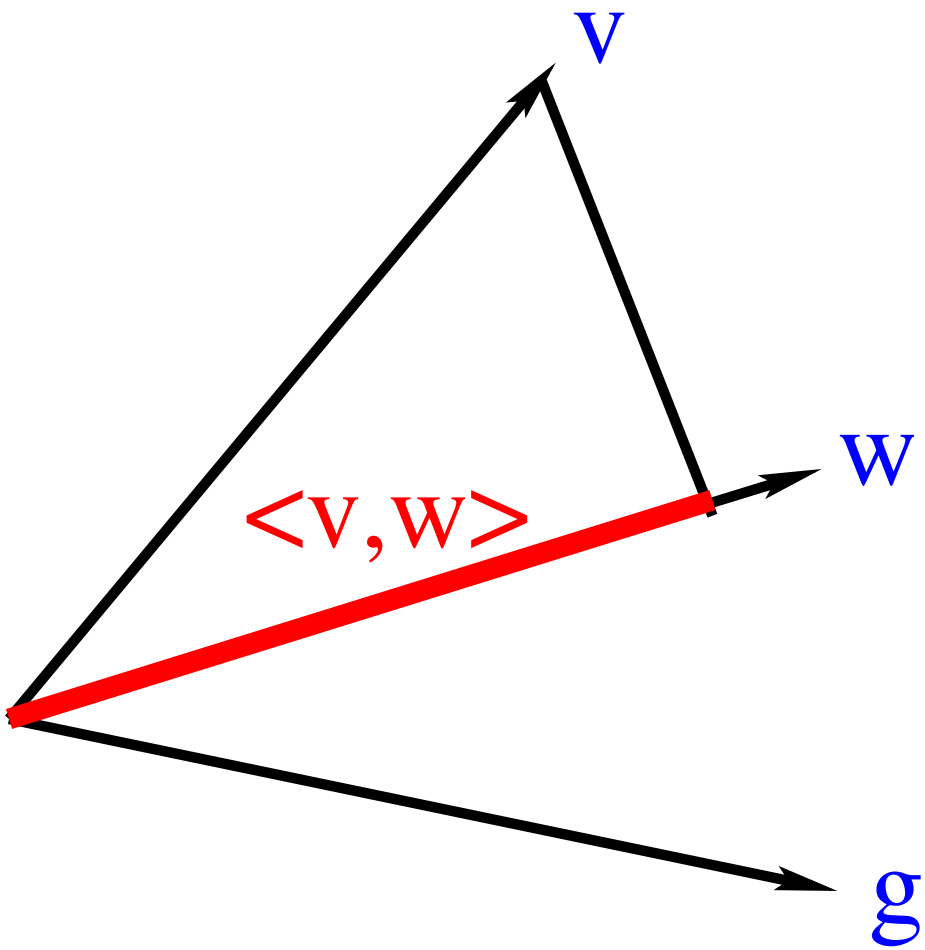
$$\sum_j T_{kj} v_j = \frac{\sum_j g'_k w'_j v_j}{\sum_i g'_i w'_i} = g'_k \frac{\sum_j v_j w'_j}{\sum_i g'_i w'_i}$$

$$\sum_k v_k T_{kj} = \frac{\sum_k v_k g'_k w'_j}{\sum_i g'_i w'_i} = w'_j \frac{\sum_k g'_k v_k}{\sum_i g'_i w'_i}$$

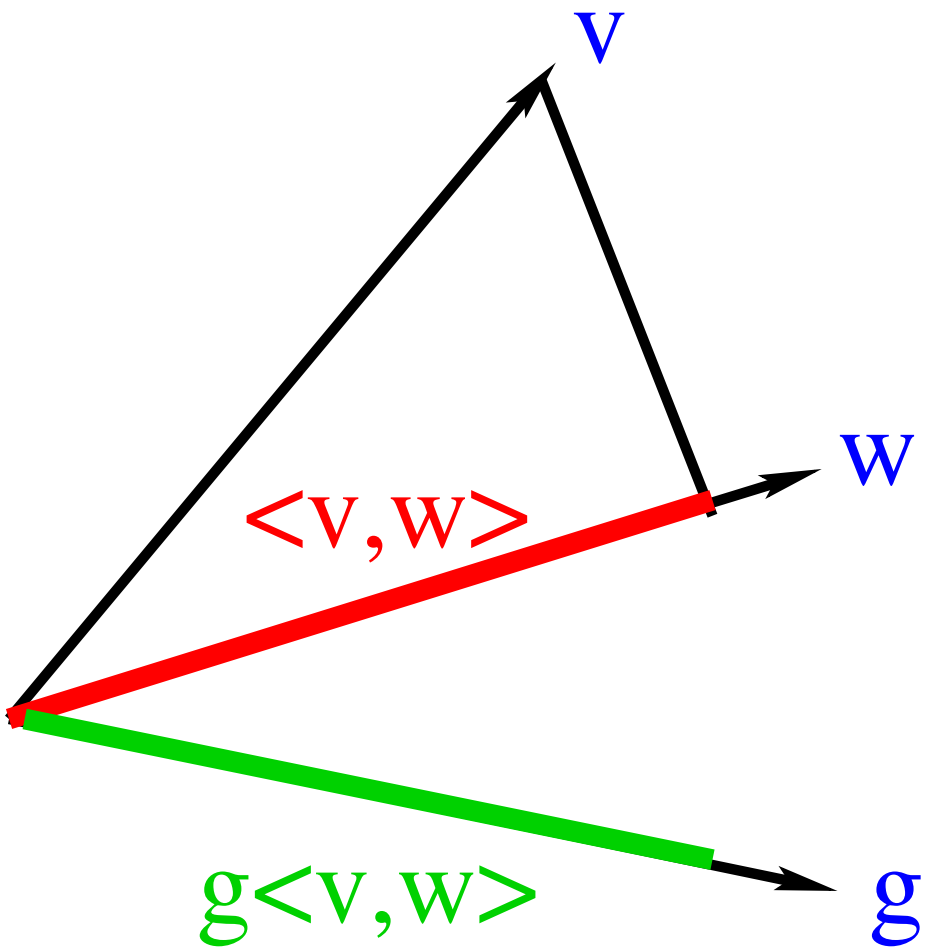
3 vectors, v, w, g



3 vectors, v, w, g



3 vectors, v, w, g



Assertion 3

Let $\delta_{kj} = I(k = j)$. if $k = j$ then 1 and 0 otherwise $S_{kj} = \delta_{kj} - T_{kj}$ Subtracts transform T_{kj} while keeping everything else.

Assertion 3

Let $\delta_{kj} = I(k = j)$. if $k = j$ then 1 and 0 otherwise $S_{kj} = \delta_{kj} - T_{kj}$ Subtracts transform T_{kj} while keeping everything else.

$$\sum_j S_{kj} v_j = v_k - g'_k \frac{\sum_j v_j w'_j}{\sum_i g'_i w'_i}$$

$$\sum_j v_k S_{kj} = v_k - w'_j \frac{\sum_k g'_k v_k}{\sum_i g'_i w'_i}$$

Assertion 4

$F_{kj} = \frac{w'_k w'_j}{\sum_i g'_i w'_i} = \frac{w' w'^{\top}}{g'^{\top} w'}$ is an estimate of the inverse Hessian.

Assertion 4

$F_{kj} = \frac{w'_k w'_j}{\sum_i g'_i w'_i} = \frac{w' w'^{\top}}{g'^{\top} w'}$ is an estimate of the inverse Hessian.

$H_{kj} = \frac{\partial^2 L}{\partial w_k \partial w_j} = \frac{\partial g_k}{\partial w_j}$

Assertion 4

$F_{kj} = \frac{w'_k w'_j}{\sum_i g'_i w'_i} = \frac{w' w'^{\top}}{g'^{\top} w'}$ is an estimate of the inverse Hessian.

$H_{kj} = \frac{\partial^2 L}{\partial w_k \partial w_j} = \frac{\partial g_k}{\partial w_j}$

So, $Hw' \simeq g'$.

Assertion 4

$F_{kj} = \frac{w'_k w'_j}{\sum_i g'_i w'_i} = \frac{w' w'^{\top}}{g'^{\top} w'}$ is an estimate of the inverse Hessian.

$H_{kj} = \frac{\partial^2 L}{\partial w_k \partial w_j} = \frac{\partial g_k}{\partial w_j}$

So, $Hw' \simeq g'$.

So an inverse should satisfy $Fg' \simeq w'$.

The BFGS direction

$D_{kj} \leftarrow \sum_{il} S_{ik} D_{il} S_{lj} + F_{kj}$

Or in recursive matrix form:

$D^t = S^{t\top} D^{t-1} S^t + F^t$

The BFGS direction

$D_{kj} \leftarrow \sum_{il} S_{ik} D_{il} S_{lj} + F_{kj}$

Or in recursive matrix form:

$D^t = {S^t}^\top D^{t-1} S^t + F^t$

Unwinding, we get:

$$D^t = {S^t}^\top {S^{t-1}}^\top \ldots {S^1}^\top D^0 S^1 S^2 \ldots S^t + {S^t}^\top \ldots {S^2}^\top F^1 S^2 \ldots S^t + \ldots + {S^t}^\top F^{t-1} S^t + F^t$$

The BFGS direction

$D_{kj} \leftarrow \sum_{il} S_{ik} D_{il} S_{lj} + F_{kj}$

Or in recursive matrix form:

$D^t = {S^t}^\top D^{t-1} S^t + F^t$

Unwinding, we get:

$$D^t = {S^t}^\top {S^{t-1}}^\top \ldots {S^1}^\top D^0 S^1 S^2 \ldots S^t + {S^t}^\top \ldots {S^2}^\top F^1 S^2 \ldots S^t + \ldots + {S^t}^\top F^{t-1} S^t + F^t$$

LBFGS is the low rank approximation.

$$L^t = {S^t}^\top \ldots {S^{t-m}}^\top D^0 S^{t-m} \ldots S^t + {S^t}^\top \ldots {S^{t-m+1}}^\top F^{t-m} S^{t-m+1} \ldots S^t + \ldots + {S^t}^\top F^{t-1} S^t + F^t$$

Questions

What is D^0?

How do you make it fast?

How do you start?

What if loss goes up?

How do you regularize?

Questions

What is D^0? $\frac{\delta_{jk}}{\frac{\partial^2 L}{\partial w_j \partial w_j}}$ is a reasonable choice.

How do you make it fast?

How do you start?

What if loss goes up?

How do you regularize?

Questions

What is D^0? $\frac{\delta_{jk}}{\frac{\partial^2 L}{\partial w_j \partial w_j}}$ is a reasonable choice.

How do you make it fast? All operations decompose into dense vector products.

How do you start?

What if loss goes up?

How do you regularize?

Questions

What is D^0? $\frac{\delta_{jk}}{\frac{\partial^2 L}{\partial w_j \partial w_j}}$ is a reasonable choice.

How do you make it fast? All operations decompose into dense vector products.

How do you start? Seed w with an online pass first.

Initially, step size may be crazy. Make a second pass computing the second derivative in the chosen direction.

What if loss goes up?

How do you regularize?

Questions

What is D^0? $\frac{\delta_{jk}}{\frac{\partial^2 L}{\partial w_j \partial w_j}}$ is a reasonable choice.

How do you make it fast? All operations decompose into dense vector products.

How do you start? Seed w with an online pass first.

Initially, step size may be crazy. Make a second pass computing the second derivative in the chosen direction.

What if loss goes up? Backstep along previous direction.

How do you regularize?

Questions

What is D^0? $\frac{\delta_{jk}}{\frac{\partial^2 L}{\partial w_j \partial w_j}}$ is a reasonable choice.

How do you make it fast? All operations decompose into dense vector products.

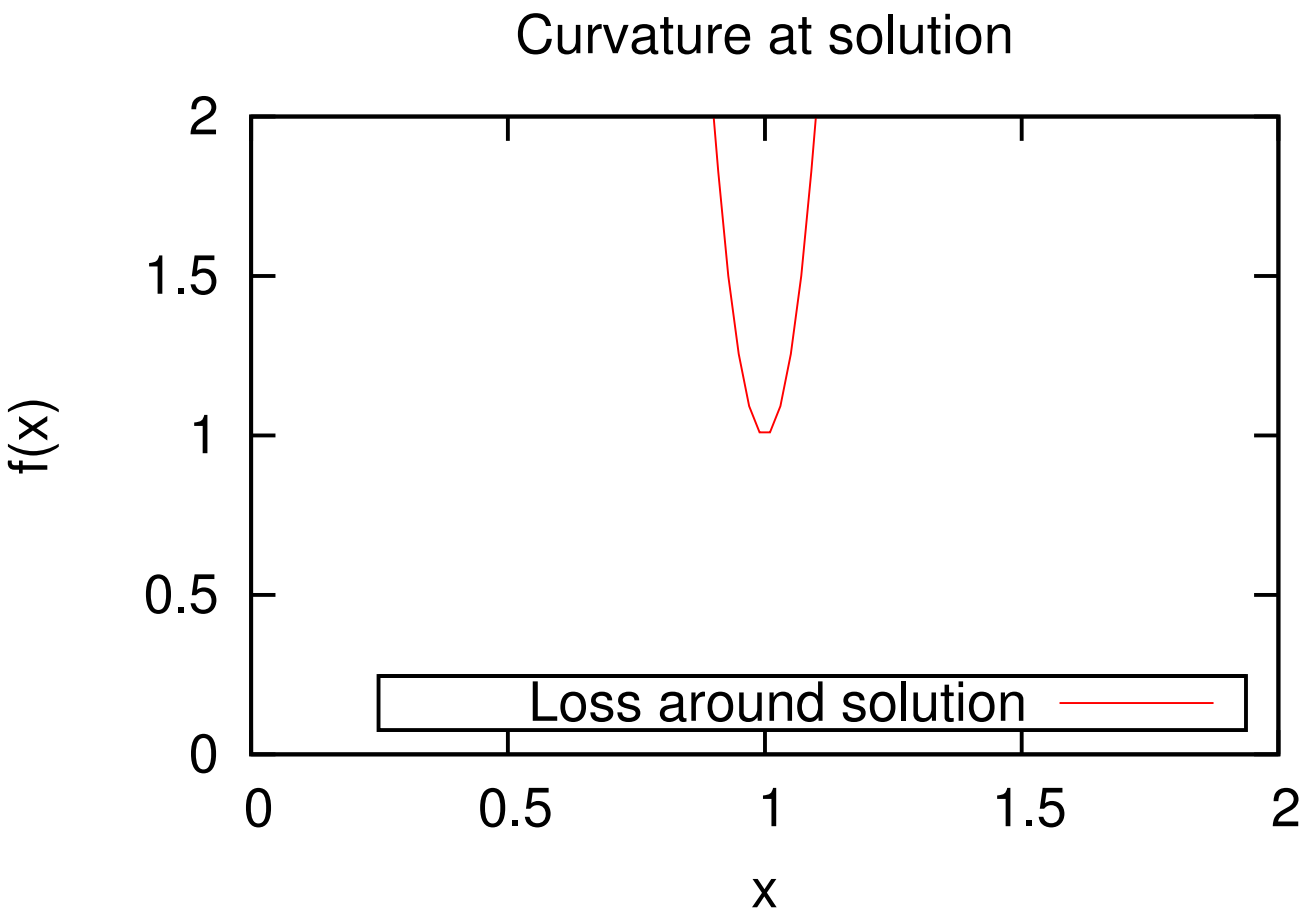
How do you start? Seed w with an online pass first. Initially, step size may be crazy. Make a second pass computing the second derivative in the chosen direction.

What if loss goes up? Backstep along previous direction.

How do you regularize? Regularized loss has the form: $L'(\hat{y}, y) = L(\hat{y}, y) + \frac{c}{2} \sum_i w_i^2$. Imposing regularization is a once-per-pass dense operation.

How do you restart with new data?

How do you restart with new data?



Compute and store: $r_i = \frac{\partial^2 L}{\partial w_i \partial w_i}$

On resumption, regularize by $\sum_i r_i (w_i - o_i)^2$ where o_i is the old weight value.

Why LBFGS?

Theorem: If L is quadratic and an exact line search was done for the step size, a variant satisfies

$$e_t \leq \frac{C}{2^{2^t}}$$

for some C.

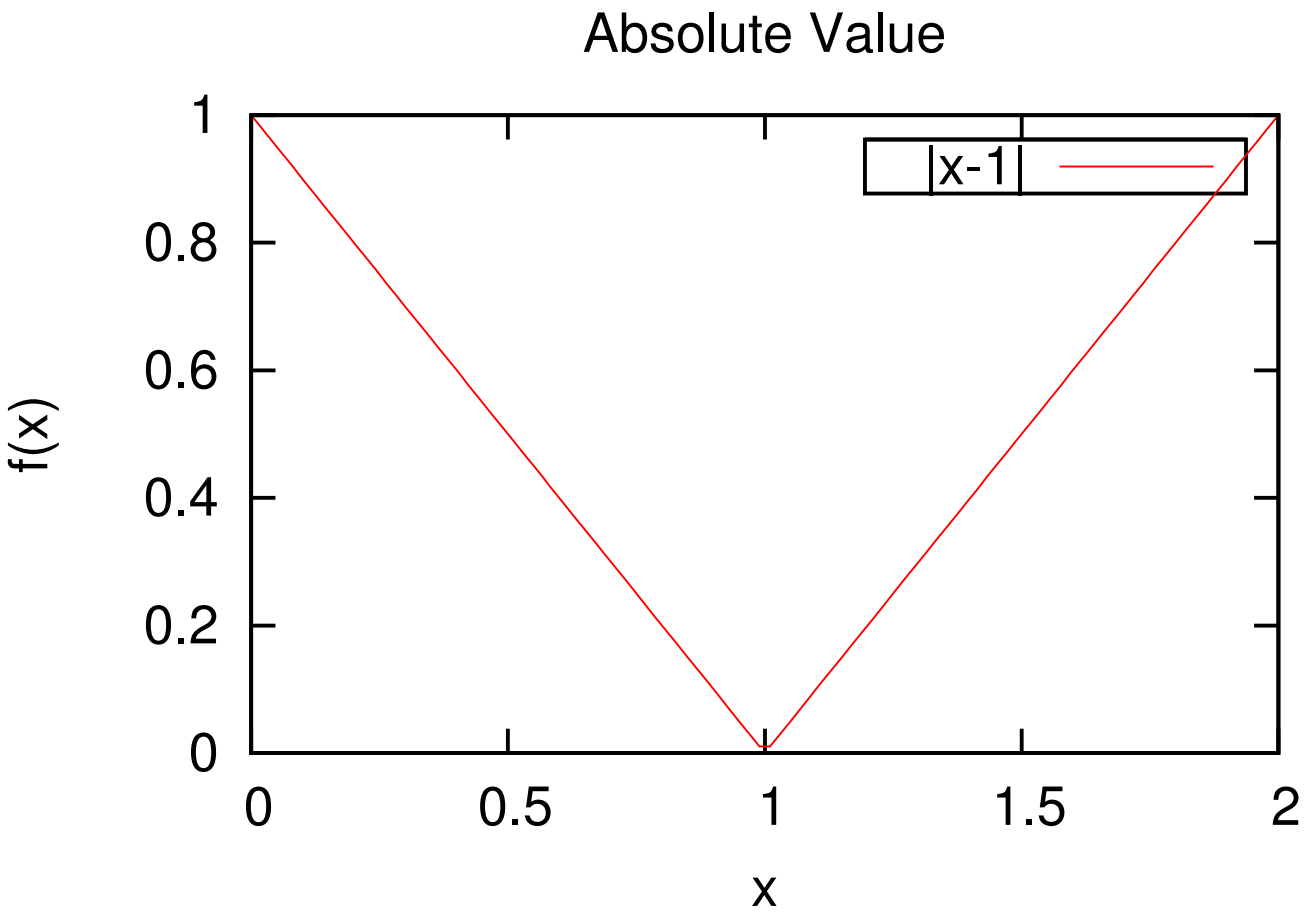
Why LBFGS?

Theorem: If L is quadratic and an exact line search was done for the step size, a variant satisfies

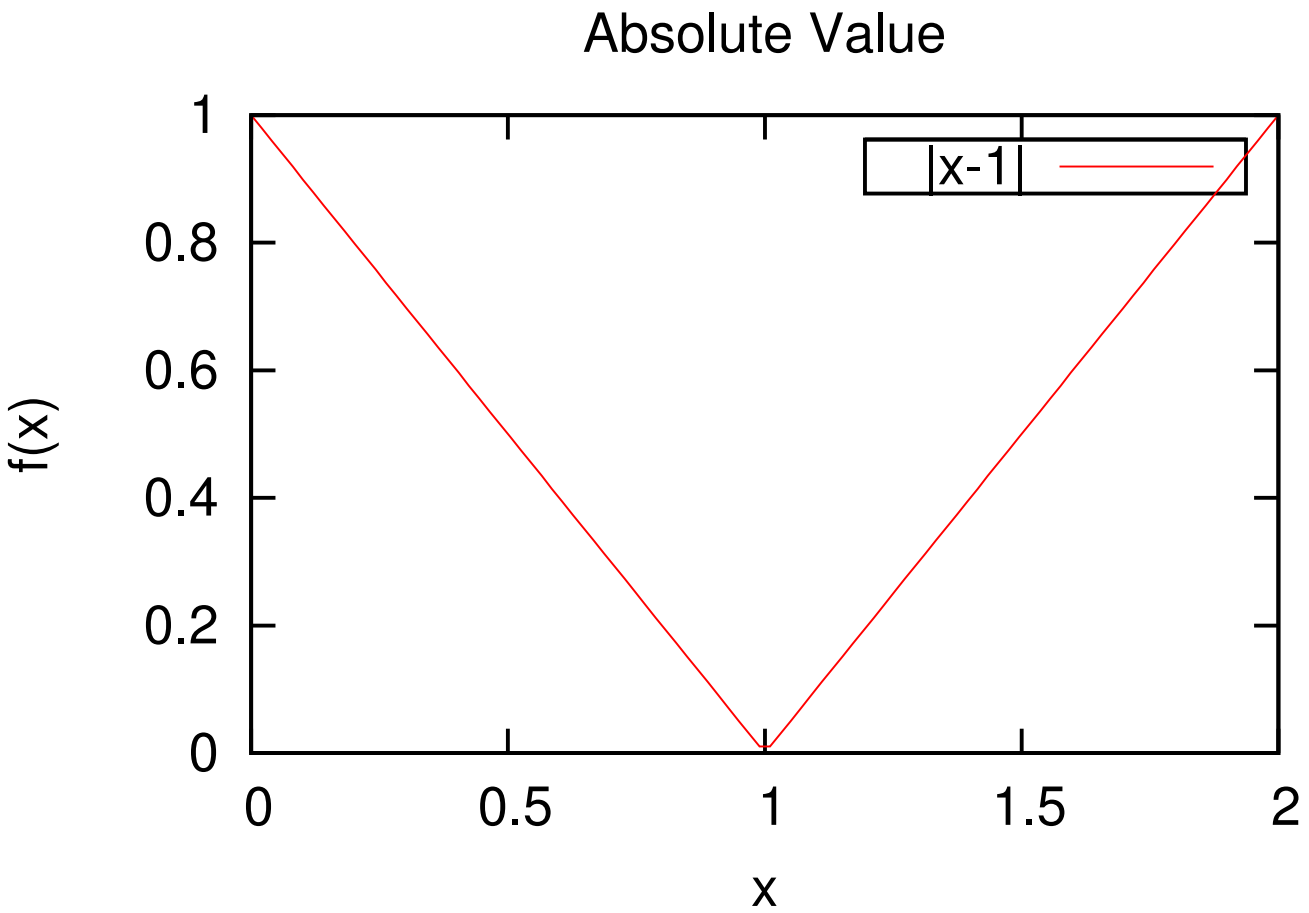
$$e_t \leq \frac{C}{2^{2^t}}$$

for some C.
Of course, it's rarely quadratic and you never perform exact line search.

What happens here?



What happens here?



What happens to a true Newton step here?

References

[L] Nocedal, J., "Updating quasi-Newton matrices with limited storage", Math. of Comp., 35, 773-782.
[B] Broyden, C., "The convergence of a class of double-rank minimization algorithms", Journal of the Inst. of Math. and Its Applications, 6:76-90.
[F] Fletcher, R., "A New Approach to Variable Metric Algorithms", Computer Journal 13 (3):317-322.
[G] Goldfarb, D., "A Family of Variable Metric Updates Derived by Variational Means", Math. of Comp. 24 (109):23-26.
[S] Shanno, D. "Conditioning of quasi-Newton methods for function minimization", Math. of Comp. 24(111):647-656.

More References

Incremental LBFGS Olivier Chapelle