Iterative Reweighted Least Squares

ECCV, Sept 7, 2014

What point minimizes the distance to the three points of a triangle?

Exercise: find this point using ruler and compass construction.
Ruler and Compass Construction to find Fermat point
Gustav Weler was a political decoy (doppelgänger or Body-double) of Adolf Hitler. At the end of the Second World War, he was executed by a gunshot to the forehead in an attempt to confuse the Allied troops when Berlin was taken. He was also used "as a decoy for security reasons".[2] When his corpse was discovered in the Reich Chancellery garden by Soviet troops, it was mistakenly believed to be that of Hitler because of his identical moustache and haircut. The corpse was also photographed and filmed by the Soviets.

One servant from the bunker declared that the dead man was one of Hitler's cooks. He also believed this man "had been assassinated because of his startling likeness to Hitler, while the latter had escaped from the ruins of Berlin".[3]

Weler's body was brought to Moscow for investigations and buried in the yard at Lefortovo prison.[4]
Andrew Vázsonyi (1916–2003), also known as Endre Weiszfeld and Zepartzatt Gozinto) was a mathematician and operations researcher. He is known for Weiszfeld's algorithm for minimizing the sum of distances to a set of points, and for founding The Institute of Management Sciences. He is known for Weiszfeld's algorithm for minimizing the sum of distances to a set of points, and for founding The Institute of Management Sciences.

E. Weiszfeld, Sur le point pour lequel la somme des distances de n points donnés est minimum, Tôhoku Mathematics Journal 43 (1937), 355 - 386.
Weiszfeld Algorithm for points

- An iterative algorithm to find $L_1$ minimum point of a set of points.
- Given a set of points $y_i$, the cost function to minimize is

$$x^* = \arg\min_x \sum_{i=1}^{k} d(x, y_i),$$

where $d(x, y_i)$ is the distance of $x$ and $y_i$.

Weiszfeld Algorithm for points

- Given points $y_i \in \mathbb{R}^N$, find the point that minimizes the $L_1$ cost function

$$C_1(x) = \sum_{i=1}^{n} d(x, y_i)$$

- Given a current estimate $x^t$, the Weiszfeld algorithm computes the next estimate $x^{t+1}$ as

$$x^{t+1} = \frac{\sum_{i=1}^{n} w_i^t y_i}{\sum_{i=1}^{n} w_i^t} = \arg\min_x \sum_{i=1}^{n} w_i^t d(x, y_i)^2$$

where $w_i^t = 1/d(x^t, y_i)$.

- $x^{t+1}$ is the centre of gravity of a configuration formed by placing a weight $w_i^t$ at each point $y_i$. 

Robust (L1) cost function

Weighted L2 cost function
Weiszfeld Algorithm for points

- Given a set of points in space. We start with a random initial estimation of the median,

Weiszfeld Algorithm for points

- Compute the sum of negative gradients
Gradient descent

- In $\mathbb{R}^n$ the cost function
  \[ C(y) = \sum_{i=1}^{n} d(x_i, y) = \sum \|x_i - y\| \]
  is convex, and has a single minimum (unless all $x_i$ are collinear).
- Gradient is
  \[ \nabla C = \sum_{i=1}^{n} (y - x_i)/\|y - x_i\| \]
- Gradient descent algorithm:
  \[ y^{t+1} = y^t + \gamma^t \sum_{i=1}^{n} (x_i - y^t)/\|x_i - y^t\| \]
  $\gamma^t$ is the step-size.
Generalizing IRLS

Functions for which IRLS works
Resistance to Outliers

Robustness to systematic outliers (e.g. ghost edges)
A general IRLS algorithm

1. Identify a weighted optimization problem that can be solved optimally (e.g. in closed form)
   \[ C(x, w) = \sum_{i=1}^{n} w_i f_i(x) \]
   Written without the squares

2. Solve iteratively: At each step, define weights (how)
   \[ w_i^t = w_i(x^t) \]
   Define weights
   and define
   \[ x^{t+1} = \text{argmin}_x C(x, w^t) \]
   Minimize weighted cost
   \[ = \text{argmin}_x \sum_{i=1}^{n} w_i^t f_i(x) \]

3. Hope that it converges to what you want.

Start with initial value \( x(0) \)

IRLS Algorithm

\[ t = 0 \]

- Define weights \( w(t) \) from \( x(t) \)
- Solve WLS problem \( x(t+1) = \text{argmin}_x C(x, w(t)) \)
- \( t = t + 1 \)
Start with initial value $x(0)$

Define weights $w(t)$ from $x(t)$

Solve WLS problem $x(t+1) = \arg\min C(x, w(t))$

$t = t+1$

IRLS Algorithm
IRLS Algorithm

Start with initial value \(x(0)\)

Define weights \(w(t)\) from \(x(t)\)

Solve WLS problem
\[x(t+1) = \text{argmin}_x C(x, w(t))\]

\(t = t+1\)
How to choose the weights

- Assume we can minimize the cost
  \[ C(x, w) = \sum_{i=1}^{n} w_i f_i(x) \]  
  \(\text{No square}!!\)

- We wish to minimize
  \[ C_h(x) = \sum_{i=1}^{n} h \circ f_i(x) \]
  \(\text{Robust cost function}\)

- We want
  \[ \nabla_x C(x, w) = 0 \text{ if and only if } \nabla_x C_h(x) = 0 \]

- So
  \[ \nabla_x w_i f_i(x) = \nabla_x (h \circ f_i(x)) \]
  \[ w_i \nabla_x f_i(x) = h'(f_i(x)) \nabla_x f_i \]
  \[ w^*_i = h'(f_i(x^*)) \]  \(\text{Required weights}\)

Example \(L_1\)

Let
\[ f_i(x) = d(x, y_i)^2 \]
\[ h(x) = \sqrt{x} \]
\[ C_h(x) = \sum_{i=1}^{n} h \circ f_i(x) = \sum_{i=1}^{n} d(x, y_i) \]
\(\text{Sum of distances}\)

Then
\[ w_i = h'(f(x)) \]
\[ = \frac{1}{2} f(x)^{-1/2} \]
\[ = \frac{1}{2} d(x, y_i)^{-1} \].
Example $L_q$

Let

\[ f_i(x) = d(x, y_i)^2 \]

\[ h(x) = x^{q/2} \]

\[ C_h(x) = \sum_{i=1}^{n} h \circ f_i(x) = \sum_{i=1}^{n} d(x, y_i)^q \]

Then

\[ w_i = h'(f(x)) \]
\[ = \frac{q}{2} f(x)^{(q-2)/2} \]

\[ w_i = \frac{q}{2} d(x, y_i)^{q-2} \]

Descent condition for IR least sum

**Lemma:** Let $h : R \rightarrow R$ be a **concave function** and let $h^s$ denote a **supergradient** of $h$. For $i = 1, \ldots, n$ let $r_i^t$ and $r_i^{t+1}$ be real numbers (residuals) such that

\[ \sum_{i=1}^{n} w_i r_i^{t+1} \leq \sum_{i=1}^{n} w_i r_i^t \]

where $w_i = h^s(r_i^t)$. Then

\[ \sum_{i=1}^{n} h(r_i^{t+1}) \leq \sum_{i=1}^{n} h(r_i^t) \]

with equality if and only if $r_i^{t+1} = r_i^t$ for all $i$. //

Apply this with $r_i^t = f_i(x^t)$ and $r_i^{t+1} = f_i(x^{t+1})$. 

Weighted residual sum decreases

Robust residual sum decreases
A concave function always has a supergradient

**Proof:** Since $h^s$ is a supergradient,

$$h(r_i^{t+1}) \leq h(r_i^t) + (r_i^{t+1} - r_i^t) h^s(r_i^t)$$

for all $i$. Summing over $i$ gives

$$\sum_{i=1}^{n} h(r_i^{t+1}) \leq \sum_{i=1}^{n} h(r_i^t) + \sum_{i=1}^{n} (r_i^{t+1} - r_i^t) h^s(r_i^t).$$

The last sum is non-positive by hypothesis, completing the proof. //

---

**General condition for descent of IRLS**

**Corollary:** Let $h : R^+ \to R$ be a function such that $h(\sqrt{x})$ is concave. For $i = 1, \ldots, n$ let $r_i^t$ and $r_i^{t+1}$ be non-negative real numbers (residuals) such that

$$\sum_{i=1}^{n} w_i (r_i^{t+1})^2 \leq \sum_{i=1}^{n} w_i (r_i^t)^2$$

where $w_i = h'(r_i^t)/r_i^t$. Then

$$\sum_{i=1}^{n} h(r_i^{t+1}) \leq \sum_{i=1}^{n} h(r_i^t)$$

with equality if and only if $r_i^{t+1} = r_i^t$ for all $i$. //
Summary

- To minimize
  \[ C_h(x) = \sum_{i=1}^{n} h \circ f_i(x) \]  
  minimize the weighted \( L_2 \) cost
  \[ C_w^2(x) = \sum_{i=1}^{n} w_i f_i(x)^2 \]  
  with weights
  \[ w_i = \frac{h'(y)}{y} \mid_{y \rightarrow f_i(x)} \]  
- Decrease in weighted \( L_2 \) cost guarantees a decrease in the robust cost, as long as:
  \[ h(\sqrt{x}) \] is concave.

Convergence

- **Warning:** Decrease in cost is no guarantee that the sequence of iterates converges!!

Where gradient descent does not converge to a minimum
Convergence conditions

If

- \( h(\sqrt{x}) \) is concave and has continuous derivative (for \( x \geq 0 \));
- \( f_i(x)^2 \) is continuously differentiable.
- \( \text{argmin}_x C_i^w(x) \) is continuous as a function of the weights \( w_i \).

then IRLS will converge to the set of critical points of \( C_i(x) \).

Hence of \( C_i(x) \) is convex, then IRLS will converge to the global minimum.

**L1**

\[
h(x) = \begin{cases} 
0 & \text{if } x < 0 \\
\frac{x}{2} & \text{if } 0 \leq x \leq 2 \\
1 & \text{if } x > 2
\end{cases}
\]

Advantage: Robust

Disadvantage:
- Function not differentiable
- Weights not defined at 0
- Can stop at non-minimum.
**Lq**

Advantage:
- Robust
- Cost function differentiable everywhere

Disadvantage:
- Weights not defined at 0
- Can stop at non-minimum.

**Huber**

Advantage:
- Robust
- Cost function differentiable
- Weights defined at zero
- Convex
- Guaranteed to converge (at least to local minimum)
Pseudo Huber

Advantage:
- Robust
- Cost function differentiable
- Weights defined at zero
- Convex
- Guaranteed to converge (at least to local minimum)

Cauchy

Advantage:
- Robust
- Cost function differentiable
- Weights defined at zero
- Guaranteed to converge (at least to local minimum)

Disadvantage:
- Non-convex
- Increased number of local minima
Blake Zisserman

\[ h(x) \]

\[ h(\sqrt{x}) \]

\[ w(y) = \frac{h'(y)}{y} \]

Advantage:
- Very robust to outliers
- Cost function differentiable
- Weights defined at zero
- Guaranteed to converge (at least to local minimum)

Disadvantage:
- Non-convex
- Increased number of local minima

Corrupted Gaussian

\[ h(x) \]

\[ h(\sqrt{x}) \]

\[ w(y) = \frac{h'(y)}{y} \]

Advantage:
- Robust
- Cost function differentiable
- Weights defined at zero
- Guaranteed to converge (at least to local minimum)

Disadvantage:
- Non-convex
- Increased number of local minima
Problems for which IRLS works

1. Any problem that you can solve the least-squares solution for exactly.
2. Point averaging.
3. Alignment of point sets (Horn’s absolute orientation problem)
4. Regression
5. Rotation averaging
6. Bundle adjustment (to local minimum)
7. ...

Example. L1 Regression
Results: $L_1$ Regression

Line Fitting:
- $L_1$ regression is compared with $L_2$ regression.

Example: Regression

- Squared distance does not work in the presence of outliers.
Example: Regression

The ideal model should be like this

Generalized Weiszfeld Algorithm

Gradient Descent:

Given a set of subspaces \( \mathcal{S}_i \), the cost function to find \( L_1 \) closest point to subspaces is

\[
\tilde{C}_2^f(X) = \sum_{i=1}^n w_i^f \| X - \mathcal{P}_{\mathcal{S}_i}(X^i) \|^2.
\]

Then, let \( X^{t+1} = \arg\min_X C_2^f(X) \).
$L_1$ closest point to subspaces

$L_1$ closest point to subspaces
$L_1$ closest point to subspaces
$L_1$ closest point to subspaces

51/63

$L_1$ closest point to subspaces

52/63
Possible application

$L_1$ optimal point of Intersection of Planes.

$L_1$ optimal point of Intersection of Planes in Aerial view.
Example. Averaging Rotations

For consistency require

\[ R_{13} = R_{23}R_{12}. \]

Define absolute rotations \( R_i \) satisfying

\[ R_{ij} = R_j R_i^{-1}. \]
Single Rotation Averaging
for Relative Orientation of Cameras

- Five corresponding points between the two images allow a computation of relative rotation (and translation).
- Very fast (about 35 μs).
- Take many different sets of 5 points and average the computed rotations.
- Individual estimates can be noisy, so we need robust method of rotation averaging.

Single Rotation Averaging

Given rotations $R_i \in SO(3)$, the $L_p$ mean is equal to

$$S^* = \arg\min_{S \in SO(3)} \sum_{i=1}^{n} d(R_i, S)^p .$$

- $p = 2$: Least-squares $L_2$ averaging. Usually simpler, not robust to outliers.
- $p = 1$: $L_1$ averaging. More robust to outliers.
So how do we average rotations?

Average of rotations is the rotation that minimizes

\[ C(S) = \sum_i d(R_i, S)^p \]

- \( p = 2 \) - Least-squares (\( L_2 \)) averaging. Usually simpler, not robust to outliers.
- \( p = 1 \) - \( L_1 \) averaging. More robust to outliers.

What is meant by \( d(R_i, S) \)?

1. Angular distance \( d_{\text{ang}}(R, S) \)
2. Quaternion distance \( \min(||r - s||, ||r + s||) \)
3. Chordal distance \( ||R - S||_F \).

Isometry of Rotations and Quaternions

Angle between two quaternions is half the angle between the corresponding rotations, defined by

\[ \angle(r_1, r_2) = \frac{\angle(R_1R_2^{-1})}{2} \]
Flatten out the meridians (longitude lines)

Azimuthal Equidistant Projection

Rotations are represented by a ball of radius $\pi$ in 3-Dimensional space.

**IRLS Algorithm on a Manifold**

- Map back and forth from the manifold to the tangent space using the exponential and logarithm maps.
**Steps of the Weiszfeld Algorithm on $SO3$**

1. Find an initial estimate $S^0$ for the median.
2. At any time $t = 0, 1, \ldots$ apply the logarithm map centred at $S^t$ to compute
   \[ v_i = \log_{S^t}(R_i). \]
3. (Weiszfeld step): Set
   \[ \delta = \frac{\sum_{i=1}^{n} v_i / \|v_i\|}{\sum_{i=1}^{n} 1 / \|v_i\|} \]
4. Set
   \[ S^{t+1} = \exp(\delta)S^t. \]
5. Repeat steps 1 to 3 until convergence.

---

**Averaging over a graph**

Relative rotations are computed between some nodes in the graph.

Initialization: Propagate rotations estimates across a tree.
Orientation of each node in turn is recomputed, given the known orientation of its neighbours, and the relative orientation.

This involves “single-rotation averaging”.

**Experimental Setup**

Notre Dame Set:
- No. of Images = 595.
- No. of 3D points = 280,000.
- No. of images pairs with >= 30 matched points = 42000.
• 569 Images
• 280,000 points
• 42,000 pairs of overlapping images (more than 30 points in common)

Task: Find the orientations of all cameras.

Extension: Optimization on Riemannian Manifolds
What is a manifold, anyways?

- Think of a manifold as a smooth surface in $\mathbb{R}^n$
- Every point on the manifold (surface) has a neighbourhood that is the same as (homeomorphic to) a ball in $\mathbb{R}^n$.

Examples of manifolds

- $\mathbb{R}^n$
- Sphere $S^n$
- Rotation space $SO(3)$ – used in rotation averaging
- Positive definite matrices – “covariance features”
- Grassman Manifolds – used to model sets of images
- Essential manifold – structure and motion
- Shape manifolds – capture the shape of an object
- Essential manifold, trifocal manifold
What is a geodesic?

A curve is a mapping $\gamma$ from an interval $[a, b]$ to the manifold $M$.

A geodesic has several descriptions:

- A **locally** distance-minimizing curve. The curve can be broken up into sections $[a_i, b_i]$ so that $\gamma$ is the shortest curve from $\gamma(a_i)$ to $\gamma(b_i)$.
- A curve on a surface whose acceleration is always normal to the surface.
- A taut piece of elastic band on the surface.
**Riemannian Manifold**

- A manifold with an inner product defined in the tangent space at each point.
- Allows us to measure angles at a point
- Define the length of curves.
- Define “geodesic distance” on the manifold
- Find curves of shortest distance.
- Define “geodesics” or locally shortest curves

**Geodesics and the exponential map**

- Exponential map wraps vector in tangent space onto the manifold.
- Constant velocity.
- Acceleration always normal to the surface.
IRLS Algorithm on a Manifold

- Map back and forth from the manifold to the tangent space using the exponential and logarithm maps.
Wesiszfeld algorithm on Manifold.

- Start with a random point on manifold

Wesiszfeld algorithm on Manifold.

- Project the points to the tangent space
Wesiszfeld algorithm on Manifold.

- Apply the Wesiszfeld algorithm

Wesiszfeld algorithm on Manifold.

- Project the updated point to the manifold
Wesiszfeld algorithm on Manifold.

- Again, project all the points on manifold to the tangent space and repeat the same procedure until convergence.

Convergence of IRLS on Manifolds

Weiszfeld algorithm will converge on a manifold of non-negative curvature (Fletcher 2009, Aftab 2011, 2014)

Why positive curvature? Toponogov's Theorem.

- With non-negative (sectional) curvature, geodesics converge.
- Distance in the tangent space is always greater than distance on the manifold.
- If iteration causes distances to decrease in the tangent space, they decrease even more in the manifold.

Hyperbolic (negative curvature) manifold