Dynamic $\ell_1$ Reconstruction

Justin Romberg
Georgia Tech, ECE

Collaborators:
M. Salman Asif
Aurèle Balavoine
Chris Rozell

Tsinghua University
October 18, 2013
Beijing, China
Goal: a dynamical framework for sparse recovery

Given $y$ and $\Phi$, solve

$$\min_x \lambda \|x\|_1 + \frac{1}{2} \|\Phi x - y\|_2^2$$
Goal: a dynamical framework for sparse recovery

We want to move from:

\[ \min_x \lambda \|x\|_1 + \frac{1}{2} \|\Phi x - y\|_2^2 \]

*Given \( y \) and \( \Phi \), solve*

\[ y(t) \overset{\min \ell_1}{\left\{ \Phi(t) \right\}} \hat{x}(t) \]

...
We will look at dynamical reconstruction in two different contexts:

- Fast updating of solutions of $\ell_1$ optimization programs

  M. Salman Asif

- Systems of nonlinear differential equations that solve $\ell_1$ (and related) optimization programs, implemented as continuous-time neural nets

  Aurèle Balavoine  
  Chris Rozell
Classical: Recursive least-squares

- System model:
  \[ y = \Phi x \]
  - \( \Phi \) has full column rank
  - \( x \) is arbitrary

- Least-squares estimate:
  \[
  \min_{\hat{x}} \| y - \Phi \hat{x} \|_2^2 \implies \hat{x} = (\Phi^T \Phi)^{-1} \Phi^T y
  \]
Classical: Recursive least-squares

- Sequential measurement:
  \[
  \begin{bmatrix}
  y \\
  w
  \end{bmatrix} = \begin{bmatrix}
  \Phi \\
  \phi^T
  \end{bmatrix} x
  \]

- Compute new estimate using rank-1 update:
  \[
  \hat{x}_1 = (\Phi^T\Phi + \phi\phi^T)^{-1}(\Phi^T y + \phi \cdot w)
  \]
  \[
  = \hat{x}_0 + K_1 (w - \phi^T x_0)
  \]
  where
  \[
  K_1 = (\Phi^T\Phi)^{-1}\phi(1 + \phi^T(\Phi^T\Phi)^{-1}\phi)^{-1}
  \]

- With the previous inverse in hand, the update has the cost of a few matrix-vector multiplies
Classical: The Kalman filter

- Linear dynamical system for state evolution and measurement:

\[
y_t = \Phi_t x_t + e_t \\
x_{t+1} = F_t x_t + f_t
\]

\[
\begin{bmatrix}
I & 0 & 0 & 0 & \cdots \\
\Phi_1 & 0 & 0 & 0 & \cdots \\
-F_1 & I & 0 & 0 & \cdots \\
0 & \Phi_2 & 0 & 0 & \cdots \\
0 & -F_2 & I & 0 & \cdots \\
0 & 0 & \Phi_3 & 0 & \cdots \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3 \\
\vdots \\
\end{bmatrix}
= 
\begin{bmatrix}
F_0 x_0 \\
y_1 \\
y_2 \\
y_3 \\
\vdots \\
\end{bmatrix}
\]

- As time marches on, we add both rows and columns.

- Least-squares problem:

\[
\min_{x_1, x_2, \ldots} \sum_t \left( \sigma_t \| \Phi_t y_t - y_t \|_2^2 + \lambda_t \| x_t - F_{t-1} x_{t-1} \|_2^2 \right)
\]
Classical: The Kalman filter

- Linear dynamical system for state evolution and measurement:

\[ y_t = \Phi_t x_t + e_t \]
\[ x_{t+1} = F_t x_t + d_t \]

- Least-squares problem:

\[ \min_{x_1, x_2, \ldots} \sum_t \left( \sigma_t \| \Phi_t y_t - y_t \|^2_2 + \lambda_t \| x_t - F_{t-1} x_{t-1} \|^2_2 \right) \]

- Again, we can use low-rank updating to solve this recursively:

\[ v_k = F_k \hat{x} \]
\[ K_{k+1} = (F_k P_k F_k^T + I) \Phi_{k+1}^T (\Phi_{k+1} (F_k P_k F_k^T + I) \Phi_{k+1}^T + I)^{-1} \]
\[ \hat{x}_{k+1|k+1} = v_k + K_{k+1} (y_{k+1} - \Phi_{k+1} v_k) \]
\[ P_{k+1} = (I - K_{k+1} \Phi_{k+1}) (F_k P_k F_k^T + I) \]
Dynamic sparse recovery: $\ell_1$ filtering

- Goal: efficient updating for optimization programs like
  \[
  \min_x \|W x\|_1 + \frac{1}{2} \|\Phi x - y\|_2^2
  \]

- We want to *dynamically update* the solution when
  - the underlying signal changes slightly,
  - we add/remove measurements,
  - the weights changes,
  - we have streaming measurements for an evolving signal (adding/removing columns from $\Phi$)
Optimality conditions for BPDN

\[
\min_x \| W x \|_1 + \frac{1}{2} \| \Phi x - y \|_2^2
\]

- Conditions for \( x^* \) (supported on \( \Gamma^* \)) to be a solution:

  \[
  \phi^T_\gamma (\Phi x^* - y) = -W[\gamma, \gamma]z[\gamma] \quad \gamma \in \Gamma^*
  \]

  \[
  |\phi^T_\gamma (\Phi x^* - y)| \leq W[\gamma, \gamma] \quad \gamma \in \Gamma^{*c}
  \]

  where \( z[\gamma] = \text{sign}(x[\gamma]) \)

- Derived simply by computing the subgradient of the functional above
Example: time-varying sparse signal

- Initial measurements. Observe
  \[ y_1 = \Phi x_1 + e_1 \]

- Initial reconstruction. Solve
  \[ \min_x \lambda \| x \|_1 + \frac{1}{2} \| \Phi x - y_1 \|_2^2 \]

We can gradually move from the first solution to the second solution using homotopy

\[ \min_x \lambda \| x \|_1 + \frac{1}{2} \| \Phi x - (1 - \epsilon) y_1 - \epsilon y_2 \|_2^2 \]

Take \( \epsilon \) from 0 \( \rightarrow \) 1
Example: time-varying sparse signal

- Initial measurements. Observe
  \[ y_1 = \Phi x_1 + e_1 \]

- Initial reconstruction. Solve
  \[ \min_x \lambda \|x\|_1 + \frac{1}{2} \|\Phi x - y_1\|_2^2 \]

- A new set of measurements arrives:
  \[ y_2 = \Phi x_2 + e_2 \]

- Reconstruct again using \( l_1 \)-min:
  \[ \min_x \lambda \|x\|_1 + \frac{1}{2} \|\Phi x - y_2\|_2^2 \]
Example: time-varying sparse signal

- Initial measurements. Observe
  \[ y_1 = \Phi x_1 + e_1 \]

- Initial reconstruction. Solve
  \[ \min_x \lambda \|x\|_1 + \frac{1}{2} \| \Phi x - y_1 \|_2^2 \]

- A new set of measurements arrives:
  \[ y_2 = \Phi x_2 + e_2 \]

- Reconstruct again using $\ell_1$-min:
  \[ \min_x \lambda \|x\|_1 + \frac{1}{2} \| \Phi x - y_2 \|_2^2 \]

- We can gradually move from the first solution to the second solution using homotopy
  \[ \min \lambda \|x\|_1 + \frac{1}{2} \| \Phi x - (1 - \epsilon)y_1 - \epsilon y_2 \|_2^2 \]

Take $\epsilon$ from $0 \to 1$
Example: time-varying sparse signal

\[
\min \; \lambda \|x\|_1 + \frac{1}{2} \|\Phi x - (1 - \epsilon)y_{\text{old}} - \epsilon y_{\text{new}}\|_2^2, \quad \text{take } \epsilon \text{ from } 0 \to 1
\]

- Path from old solution to new solution is \textit{piecewise linear}
- Optimality conditions for fixed \( \epsilon \):

\[
\Phi^T_\Gamma (\Phi x - (1 - \epsilon)y_{\text{old}} - \epsilon y_{\text{new}}) = -\lambda \text{ sign } x_\Gamma
\]

\[
\|\Phi^T_{\Gamma_{\text{c}}} (\Phi x - (1 - \epsilon)y_{\text{old}} - \epsilon y_{\text{new}})\|_\infty < \lambda
\]

\( \Gamma = \text{active support} \)
- Update direction:

\[
\partial x = \begin{cases} 
-(\Phi^T_\Gamma \Phi_\Gamma)^{-1}(y_{\text{old}} - y_{\text{new}}) & \text{on } \Gamma \\
0 & \text{off } \Gamma
\end{cases}
\]
Path from old solution to new

Γ = support of current solution.

Move in this direction

\[ \partial x = \begin{cases} 
- (\Phi_T \Phi) \Gamma^{-1} (y_{old} - y_{new}) & \text{on } \Gamma \\
0 & \text{off } \Gamma 
\end{cases} \]

until support changes, or one of these constraints is violated:

\[ \left| \phi^T_\gamma (\Phi (x + \epsilon \partial x) - (1 - \epsilon) y_{old} - \epsilon y_{new}) \right| < \lambda \quad \text{for all } \gamma \in \Gamma^c \]
Sparse innovations

House

Blocks

Pcw. poly

Piecewise constant signal [adapted from WaveLab]

Piecewise polynomial signal (cubic) [adapted from WaveLab]

Zoom in for Haar wavelet coefficients

Zoom in for wavelet coefficients (using Daub8)
**Numerical experiments: time-varying sparse signals**

<table>
<thead>
<tr>
<th>Signal type</th>
<th>DynamicX* (nProdAtA, CPU)</th>
<th>LASSO homotopy (nProdAtA, CPU)</th>
<th>GPSR-BB (nProdAtA, CPU)</th>
<th>FPC_AS (nProdAtA, CPU)</th>
</tr>
</thead>
<tbody>
<tr>
<td>N = 1024</td>
<td>(23.72, 0.132)</td>
<td>(235, 0.924)</td>
<td>(104.5, 0.18)</td>
<td>(148.65, 0.177)</td>
</tr>
<tr>
<td>M = 512</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T = m/5, k ~ T/20</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Values = +/- 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Blocks</td>
<td>(2.7, 0.028)</td>
<td>(76.8, 0.490)</td>
<td>(17, 0.133)</td>
<td>(53.5, 0.196)</td>
</tr>
<tr>
<td>Pcw. Poly.</td>
<td>(13.83, 0.151)</td>
<td>(150.2, 1.096)</td>
<td>(26.05, 0.212)</td>
<td>(66.89, 0.25)</td>
</tr>
<tr>
<td>House slices</td>
<td>(26.2, 0.011)</td>
<td>(53.4, 0.019)</td>
<td>(92.24, 0.012)</td>
<td>(90.9, 0.036)</td>
</tr>
</tbody>
</table>

\[ \tau = 0.01 \| A^T y \|_\infty \]

nProdAtA: roughly the avg. no. of matrix vector products with \( A \) and \( A^T \)

CPU: average cputime to solve

[Asif and R. 2009]
Adding a measurement

- Analog of recursive least squares for $\ell_1$ min:

\[
\begin{bmatrix}
  y \\
  w
\end{bmatrix} = \begin{bmatrix}
  \Phi \\
  \phi
\end{bmatrix} x + \begin{bmatrix}
  e \\
  d
\end{bmatrix} \quad \rightarrow \quad \min_x \lambda \|x\|_1 + \frac{1}{2} \|\Phi x - y\|_2^2 + \frac{1}{2} \|\phi, x\| - w\|^2
\]

- Work in the new measurement slowly

\[
\min \lambda \|x\|_1 + \frac{1}{2} \left( \|\Phi x - y\|_2^2 + \epsilon \|\phi, x\| - w\|^2 \right)
\]

Again, the solution path is piecewise linear in $\epsilon$

[Garrigues et al. 08, Asif and R 09]
Adding a measurement: updating

- **Optimality conditions**

\[
\Phi^T_\Gamma (\Phi x - y) + \epsilon (\langle \phi, x \rangle - w) \phi_\Gamma = -\lambda \text{sign} x_\Gamma \\
\|\Phi^{T}_{\Gamma c} (\Phi x - y) + \epsilon (\langle \phi, x \rangle - w) \phi_{\Gamma c}\|_\infty < \lambda
\]

- **From critical point** \(x_{\epsilon_0}\), update direction is

\[
\partial x = \begin{cases} 
(w - \langle \phi, x_{\epsilon_0} \rangle) \cdot (\Phi^T_\Gamma \Phi_\Gamma + \epsilon_0 \phi \phi^T)^{-1} \phi_\Gamma & \text{on } \Gamma \\
0 & \text{off } \Gamma
\end{cases}
\]
Numerical experiments: adding a measurement

\[ N = 1024, \text{ measurements } M = 512, \text{ sparsity } S = 100 \]

Add \( P \) new measurements

Compare the average number of \textit{matrix-vector} products per update

<table>
<thead>
<tr>
<th>( P )</th>
<th>( \lambda ) [ (\tau = \lambda | A^T y |_\infty) ]</th>
<th>DynamicSeq</th>
<th>LASSO</th>
<th>GPSR-BB</th>
<th>FPC_AS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.5</td>
<td>2.3</td>
<td>41.86</td>
<td>11.86</td>
<td>15.98</td>
</tr>
<tr>
<td></td>
<td>0.1</td>
<td>4.72</td>
<td>159.76</td>
<td>42.64</td>
<td>50.70</td>
</tr>
<tr>
<td></td>
<td>0.05</td>
<td>4.5</td>
<td>162.34</td>
<td>38.80</td>
<td>97.73</td>
</tr>
<tr>
<td></td>
<td>0.01</td>
<td>8.02</td>
<td>233.70</td>
<td>55.46</td>
<td>79.83</td>
</tr>
<tr>
<td>5</td>
<td>0.5</td>
<td>5.88</td>
<td>42.00</td>
<td>14.24</td>
<td>15.96</td>
</tr>
<tr>
<td></td>
<td>0.1</td>
<td>9.58</td>
<td>152.54</td>
<td>46.42</td>
<td>47.48</td>
</tr>
<tr>
<td></td>
<td>0.05</td>
<td>10.70</td>
<td>161.36</td>
<td>47.96</td>
<td>98.75</td>
</tr>
<tr>
<td></td>
<td>0.01</td>
<td>20.32</td>
<td>227.82</td>
<td>66.64</td>
<td>78.58</td>
</tr>
<tr>
<td>10</td>
<td>0.5</td>
<td>7.6</td>
<td>44.72</td>
<td>14.96</td>
<td>16.12</td>
</tr>
<tr>
<td></td>
<td>0.1</td>
<td>14.98</td>
<td>155.26</td>
<td>53.12</td>
<td>47.05</td>
</tr>
<tr>
<td></td>
<td>0.05</td>
<td>16.40</td>
<td>162.72</td>
<td>52.12</td>
<td>98.51</td>
</tr>
<tr>
<td></td>
<td>0.01</td>
<td>29.34</td>
<td>241.52</td>
<td>75.44</td>
<td>82.91</td>
</tr>
</tbody>
</table>
Reweighted $\ell_1$

Weighted $\ell_1$ reconstruction:

$$\min_x \sum_k w_k |x_k| + \frac{1}{2} \|\Phi x - y\|_2^2 = \min_x \|Wx\|_1 + \frac{1}{2} \|\Phi x - y\|_2^2$$

solve this iteratively, adapting the weights to the previous solution:

$$w_k = \frac{\lambda}{|x_k^{\text{old}}| + c}$$

(from Boyd, Candes, Wakin '08)
Changing the weights

Iterative reweighting: take \( \{w_k\} \to \{\tilde{w}_k\} \)

Optimality conditions:

\[
\phi^*_k(y - \Phi x) = (\epsilon w_k + (1 - \epsilon)\tilde{w}_k)z_k \quad \text{on support, } k \in \Gamma
\]
\[
|\phi^*_k(y - \Phi x)| < \epsilon w_k + (1 - \epsilon)\tilde{w}_k \quad \text{off support, } k \in \Gamma^c
\]

Update direction (increasing \( \epsilon \)):

\[
\partial x = \begin{cases} 
(\Phi^*_\Gamma \Phi_\Gamma)^{-1}(W - \tilde{W})z & \text{on } \Gamma \\
0 & \text{on } \Gamma^c
\end{cases}
\]

[Asif and R 2012]
Numerical experiments: changing the weights

Numerical Experiments

Sparse signal of length N recovered from M Gaussian measurements

- ADP-H (adaptive weighting via homotopy)
- IRW-H (iterative reweighting via homotopy)

SpaRSA [Wright et al 2007]
YALL1 [Yang et al 2007]
A general, flexible homotopy framework

- Formulations above
  - depend critically on maintaining optimality
  - are very efficient when the solutions are close
- Streaming measurements for evolving signals require some type of predict and update framework

Kalman filter: \( v_k = F \hat{x}_k \), \hspace{1cm} (predict)
\[ \hat{x}_{k+1} = v_k + K(y - \Phi_k v_k), \] \hspace{1cm} (update)

- What program does the prediction \( v_k \) solve?
- Can we trace the path to the solution from a general “warm start”?
A general, flexible homotopy framework

We want to solve

$$\min_x \|Wx\|_1 + \frac{1}{2}\|\Phi x - y\|_2^2$$

- Initial guess/prediction: $v$
- Solve

$$\min_x \|Wx\|_1 + \frac{1}{2}\|\Phi x - y\|_2^2 + (1 - \epsilon)u^T x$$

for $\epsilon : 0 \rightarrow 1$.
- Taking

$$u = -Wz - \Phi^T(\Phi v - y)$$

for some $z \in \partial(\|v\|_1)$ makes $v$ optimal for $\epsilon = 0$.
Moving from the warm-start to the solution

\[
\min_x \|Wx\|_1 + \frac{1}{2}\|\Phi x - y\|_2^2 + (1 - \epsilon)u^T x
\]

The optimality conditions are

\[
\Phi^T_{\Gamma}(\Phi x - y) + (1 - \epsilon)u = -W \text{sign } x_{\Gamma} \\
|\phi^T_{\gamma}(\Phi x - y) + (1 - \epsilon)u| \leq W[\gamma, \gamma]
\]

We move in direction

\[
\partial x = \begin{cases} 
  u_{\Gamma} & \text{on } \Gamma \\
  0 & \text{on } \Gamma^c
\end{cases}
\]

until a component shrinks to zero or a constraint is violated, yielding new \( \Gamma \)
Streaming basis: Lapped orthogonal transform

- Streaming signal recovery - Results
- (Top-left) Mishmash signal (zoomed in for first 2560 samples.
- (Top-right) Error in the reconstruction at $R = N/M = 4$.
- (Bottom-left) LOT coefficients. (Bottom-right) Error in LOT coefficients.
Streaming sparse recovery

Observations: \[ y_t = \Phi_t x_t + e_t \]

Representation: \[ x[n] = \sum_{p,k} \alpha_{p,k} \psi_{p,k}[n] \]
Streaming sparse recovery

Iteratively reconstruct the signal over a sliding (active) interval, form $u$ from your prediction, then take $\epsilon : 0 \rightarrow 1$ in

$$
\min_{\alpha} \| W \alpha \|_1 + \frac{1}{2} \| \Phi \tilde{\Psi} \alpha - \tilde{y} \|_2^2 + (1 - \epsilon) u^T \alpha
$$

where $\tilde{\Psi}, \tilde{y}$ account for edge effects

$$
\tilde{\Phi} \tilde{y} = \tilde{\alpha} + \tilde{e}
$$

Divide the system into two parts

$$
\tilde{\Phi} \tilde{\Psi} \tilde{\alpha} + \tilde{\Psi} \tilde{\alpha} + \tilde{e} \Rightarrow \tilde{y} = \tilde{y} - \tilde{\Phi} \tilde{\Psi} \tilde{\alpha}
$$
Streaming signal recovery: Simulation

(Top-left) Mishmash signal (zoomed in for first 2560 samples.
(Top-right) Error in the reconstruction at $R=N/M = 4$.
(Bottom-left) LOT coefficients. (Bottom-right) Error in LOT coefficients
Streaming signal recovery: Simulation

(Left) SER at different R from ±1 random measurements in 35 db noise
(Middle) Count for matrix-vector multiplications
(Right) Matlab execution time
Streaming signal recovery: Dynamic signal

Observation/evolution model:

\[ y_t = \Phi_t x_t + e_t \]
\[ x_{t+1} = F_t x_t + d_t \]

We solve

\[
\min_{\alpha} \sum_t \| W_t \alpha_t \|_1 + \frac{1}{2} \| \Phi_t \Psi_t \alpha_t - y_t \|_2^2 + \frac{1}{2} \| F_{t-1} \Psi_{t-1} \alpha_{t-1} - \Psi_t \alpha_t \|_2^2
\]

(formulation similar to Vaswani 08, Carmi et al 09, Angelosante et al 09, Zainel at al 10, Charles et al 11)

using

\[
\min_{\alpha} \| W \alpha \|_1 + \frac{1}{2} \| \Phi \Psi \alpha - \tilde{y} \|_2^2 + \frac{1}{2} \| \tilde{F} \Psi \alpha - \tilde{q} \|_2^2 + (1 - \epsilon) u^T \alpha
\]
Dynamic signal: Simulation

(Top-left) Piece-regular signal (shifted copies) in image
(Top-right) Error in the reconstruction at R=N/M = 4.
(Bottom-left) Reconstructed signal at R=4.
(Bottom-right) Comparison of SER for the L1-regularized and the L2-regularized problems
Dynamic signal: Simulation

(left) SER at different R from ±1 random measurements in 35 db noise
(middle) Count for matrix-vector multiplications
(right) Matlab execution time

(1) SER at different R from ±1 random measurements in 35 db noise
(middle) Count for matrix-vector multiplications
(right) Matlab execution time
Dynamical systems for sparse recovery
Approximate analog computing

- Radical re-think of how computer arithmetic is done — computations use the physics of the devices (transistors) more directly
- Use < 1% of the transistors, maybe 1/10,000 of the power, possibly 100x faster than GPU
- Computations are noisy, overall precision $\approx 10^{-2}$
Approximate analog computing

• Radical re-think of how computer arithmetic is done — computations use the physics of the devices (transistors) more directly
• Use $< 1\%$ of the transistors, maybe $1/10,000$ of the power, possibly $100\times$ faster than GPU
• Computations are noisy, overall precision $\approx 10^{-2}$
• Small scale successes (embedded beamforming, adaptive filtering)
Approximate analog computing

- Radical re-think of how computer arithmetic is done — computations use the physics of the devices (transistors) more directly
- Use < 1% of the transistors, maybe 1/10,000 of the power, possibly 100x faster than GPU
- Computations are *noisy*, overall precision $\approx 10^{-2}$
- Small scale successes (embedded beamforming, adaptive filtering)
- Medium to large scale potential
  - FPAAs
  - specialized circuits for optimization
    - Hopfield networks, neural net implementations
  - general (SIMD) computing architecture ?
- Much of this work is proprietary, start-ups swallowed up by AD, NI, ...
  - Lyric semiconductor, GTronix, Singular Computing, ...
Analog vector-matrix-multiply

- Digital Multiply-and-Accumulate
  - Small time constant
  - Low power consumption

- Analog Vector-Matrix Multiplier
  - Limited accuracy
  - Limited dynamic range
Dynamical systems for sparse recovery

There are simple systems of nonlinear differential equations that settle to the solution of

$$\min_x \lambda \|x\|_1 + \frac{1}{2} \|\Phi x - y\|_2^2$$

or more generally

$$\min_x \lambda \sum_{n=1}^{N} C(x[n]) + \frac{1}{2} \|\Phi x - y\|_2^2$$

The Locally Competitive Algorithm (LCA):

$$\tau \dot{u}(t) = -u(t) - (\Phi^T \Phi - I)x(t) + \Phi^T y$$

$$x(t) = T_\lambda(u(t))$$

is a neurologically-inspired (Rozell et al 08) system which settles to the solutions of the above
Locally competitive algorithm

\[ \tau \dot{u}(t) = -u(t) - (\Phi^T \Phi - I)x(t) + \Phi^T y \]

\[ x(t) = T_\lambda(u(t)) \]
Locally competitive algorithm

Cost function

\[ V(x) = \lambda \sum_n C(x_n) + \frac{1}{2} \| \Phi x - y \|_2^2 \]
\[ \tau \dot{u}(t) = -u(t) - (\Phi^T \Phi - I) x(t) + \Phi^T y \]
\[ x_n(t) = T_\lambda(u_n(t)) \]

\[ C(u_n) \quad \lambda \frac{dC}{dx}(u) = u - x \]
\[ x_n = T_\lambda(u_n) \]
Key questions

- Uniform convergence
- Convergence speed (general)
- Convergence speed for sparse recovery via $\ell_1$ minimization

\[
\min_x \lambda \sum_n C(x_n) + \frac{1}{2} \| \Phi x - y \|_2^2
\]
LCA convergence

Assumptions

1. \( u - a \in \lambda \partial C(a) \)

2. \( x = T_\lambda(u) = \begin{cases} 0 & |u| \leq \lambda \\ f(u) & |u| > \lambda \end{cases} \)

3. \( T_\lambda(\cdot) \) is odd and continuous, \( f'(u) > 0, \ f(u) < u \)
LCA convergence

Global asymptotic convergence:

If 1–3 hold above, then the outputs stop moving eventually:

\[ \dot{x}(t) \to 0 \quad \text{as} \quad t \to \infty \]

If the critical points of the cost function are isolated then

\[ x(t) \to x^*, \quad u(t) \to u^*, \quad \text{as} \quad t \to \infty \]
LCA convergence

Assumptions

1. \( u - a \in \lambda \partial C(a) \)

2. \( x = T_{\lambda}(u) = \begin{cases} 
0 & |u| \leq \lambda \\
\frac{f(u)}{u} & |u| > \lambda 
\end{cases} \)

3. \( T_{\lambda}(\cdot) \) is odd and continuous, \( f'(u) > 0, \ f(u) < u \)

4. \( f(\cdot) \) is subanalytic

5. \( f'(u) \leq \alpha \)
Global asymptotic convergence:

If 1–5 hold above, then the LCA is globally asymptotically convergent:

\[ x(t) \rightarrow x^*, \quad u(t) \rightarrow u^*, \quad \text{as } t \rightarrow \infty \]

where \( x^* \) is a critical point of the functional.
Convergence: support is recovered in finite time

If the LCA converges to a fixed point $u^*$ such that

$$|u_\gamma| \geq \lambda + r, \text{ and } |u_\gamma| \leq \lambda - r$$

for all $\gamma \in \Gamma^c$, then the support of $x^*$ is recovered in finite time.

$\Phi = [DCT \ I]$

$M = 256, \ N = 512$
In addition to the conditions for global convergence, if there exists $0 \leq \delta < 1$ such that for all $t \geq 0$

$$(1 - \delta) \|\tilde{x}(t)\|_2^2 \leq \|\Phi \tilde{x}(t)\|_2^2 \leq (1 + \delta) \|\tilde{x}(t)\|_2^2,$$

where $\tilde{x}(t) = x(t) - x^*$, and $\alpha d < 1 \ (f'(u) \leq \alpha)$, then the LCA exponentially converges to a unique fixed point:

$$\|u(t) - u^*\|_2 \leq \kappa_0 e^{-(1-\alpha\delta)/\tau}$$
In addition to the conditions for global convergence, if there exists $0 \leq \delta < 1$ such that for all $t \geq 0$

$$(1 - \delta)\|\tilde{x}(t)\|^2_2 \leq \|\Phi \tilde{x}(t)\|^2_2 \leq (1 + \delta)\|\tilde{x}(t)\|^2_2,$$

where $\tilde{x}(t) = x(t) - x^*$, and $\alpha d < 1$ ($f'(u) \leq \alpha$), then the LCA exponentially converges to a unique fixed point:

$$\|u(t) - u^*\|_2 \leq \kappa_0 e^{-(1-\alpha \delta)/\tau}$$

Of course, this depends on not too many nodes being active at any one time ...
Activation in proper subsets for $\ell_1$

If $\Phi$ a “random compressed sensing matrix” and

$$M \geq \text{Const} \cdot S^2 \log(N/S)$$

then for reasonably small values of $\lambda$ and starting from rest

$$\Gamma(t) \subset \Gamma^*$$

That is, only subsets of the true support are ever active.

Similar results for OMP and homotopy algorithms in CS literature.
Efficient activation for $\ell_1$

If $\Phi$ a “random compressed sensing matrix” and

$$M \geq \text{Const} \cdot S \log(N/S)$$

then for reasonably small values of $\lambda$ and starting from rest

$$|\Gamma(t)| \leq 2|\Gamma^*|$$

Similar results for OMP/ROMP, CoSAMP, etc. in CS literature
References


http://users.ece.gatech.edu/~justin/Publications.html