Reverse Engineering Dynamical Systems From Time-Series Data Using $\ell_1$-Minimization

Robert Thompson

ASU School of Mathematical & Statistical Sciences

November 2011
1. Brief Basis Pursuit Review
2. Sparse Polynomial Identification
3. ODE System Identification
4. PDE System Identification
Basis Pursuit

Find the sparsest vector that satisfies a set of constraints:

\[
\min_x \|x\|_1 \quad \text{s.t.} \quad y = Ax \quad \text{(LP)}
\]

Find the sparsest vector that comes close to satisfying a set of constraints:

\[
\min_x \|x\|_1 \quad \text{s.t.} \quad \|y - Ax\|_2 < \epsilon \quad \text{(SOCP)}
\]
Sparse Polynomial Identification: the problem

1. Given a polynomial:

\[ f(t) = \sum_{i=1}^{N} \alpha_i \phi_i(t) \quad t \in (-1, 1) \]

where \( f \) is \( K \)-sparse: \( \# \{ i : \alpha_i \neq 0 \} = K < N \), and the \( \alpha_i \) unknown.

2. Measure:

\[ y = (f(t_1), f(t_2), \ldots, f(t_m))^T \]

\[ K < m < N \]

3. Determine \( x = (\alpha_1, \ldots, \alpha_N) \).
Sparse Polynomial Identification: Basis Pursuit

\[ \begin{bmatrix} f(t_1) \\ f(t_2) \\ \vdots \\ f(t_m) \end{bmatrix}_{y \in \mathbb{R}^m} = \begin{bmatrix} \phi_1(t_1) & \phi_2(t_1) & \cdots & \phi_N(t_1) \\ \phi_1(t_2) & \phi_2(t_2) & \cdots & \phi_N(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(t_m) & \phi_2(t_m) & \cdots & \phi_N(t_m) \end{bmatrix}_{A \in \mathbb{R}^{m \times N}} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_N \end{bmatrix}_{x \in \mathbb{R}^N, \ K-\text{sparse}} \]

Apply Basis Pursuit:

\[ \min_{x} \|x\|_1 \quad \text{s.t.} \quad y = Ax \quad \rightarrow \quad f = \sum_{i=1}^{N} \alpha_i \phi_i(t) \]
“Sparse Identification”? 

- **Interpolation**: given $N + 1$ points, find the consistent polynomial of degree $N$.
- **Regression**: given $P \gg N$ points, force data to fit pre-selected model of degree $N$.
- **“Sparse Identification”**: given $m \ll N$ points, find the sparsest model that fits the data.
Sparse Polynomial Identification in Natural Basis

- $\phi_i(t) = t^{i-1}$
- $A$ is Vandermonde matrix.
- Badly conditioned!

$$y = \begin{bmatrix} f(t_1) \\ f(t_2) \\ \vdots \\ f(t_m) \end{bmatrix} = \begin{bmatrix} t_1^0 & t_1^1 & \ldots & t_1^{N-1} \\ t_2^0 & t_2^1 & \ldots & t_2^{N-1} \\ \vdots & \vdots & \ddots & \vdots \\ t_m^0 & t_m^1 & \ldots & t_m^{N-1} \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_N \end{bmatrix}$$
Sparse Polynomial Identification Experiments

1. Select random $K$-sparse $x \in \mathbb{R}^N$

2. Measure:
   \[ y = (f(t_1), \ldots, f(t_m))^T \]
   
   $t_m$ uniformly distributed in $(-1, 1)$.

3. Construct:
   \[ A_{ij} = (t_j^i - 1), \ i \in (1 \ldots m), \ j \in (1, N) \]

4. Find:
   \[ x^* = \min_{x' \in \mathbb{R}^N} \|x'\| \text{ s.t. } Ax' = y \]

5. Measure accuracy of recovery: $e = \|x^* - x\|_2$

6. $e < \mu = \text{success}$
Discovering 2D Sparse Polynomials - Recovery Example

\[ x \quad x^* \]

\[
\begin{align*}
-0.0000 & \quad 0 \\
0.0000 & \quad 0 \\
0.0000 & \quad 0 \\
-0.0000 & \quad 0 \\
0.0000 & \quad 0 \\
-0.0000 & \quad 0 \\
-0.0000 & \quad 0 \\
-0.1168 & \quad -0.1168 \\
1.2525 & \quad 1.2525 \\
0.0000 & \quad 0 \\
0.5675 & \quad 0.5675 \\
0.0000 & \quad 0 \\
-0.0000 & \quad 0 \\
\ldots & \quad \ldots \\
0.0000 & \quad 0 \\
-0.0000 & \quad 0 \\
0.0000 & \quad 0 \\
-1.5505 & \quad -1.5505 \\
0.4812 & \quad 0.4812 \\
-0.0000 & \quad 0 \\
0.0000 & \quad 0 \\
\ldots & \quad \ldots
\end{align*}
\]

\[
\downarrow \quad \text{Recovered from 25 samples on (0, 1)}
\]
How Well Does It Work?

Each pixel plots recovery rate after 50 experiments, dark $\rightarrow$ better recovery rate.

Increasing $m$ doesn’t make as much difference as might be hoped.
Increasing $m$ doesn’t help

- Many sets of columns of $V$ are nearly dependent.
- Many $K$-sparse vectors $x \in \mathbb{R}^N$ become indistinguishable once transformed by $A$ to $y \in \mathbb{R}^m$, even with $m = N$.
- Increasing $m$ can’t solve this problem.
Consider same scenario, but now using Chebyshev polynomials:

$$\phi_i(t) = T_i(t) = \cos((i - 1) \arccos(t))$$

\[
y = \begin{bmatrix} f(t_1) \\ f(t_2) \\ \vdots \\ f(t_m) \end{bmatrix} = \begin{bmatrix} T_1(t_1) & T_2(t_1) & \ldots & T_N(t_1) \\ T_1(t_2) & T_2(t_2) & \ldots & T_N(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ T_1(t_m) & T_2(t_m) & \ldots & T_N(t_m) \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_N \end{bmatrix}
\]
How Well Does It Work?

Vandermonde

Using Chebyshev basis functions, we realize improvement as $m$ increases.
Increasing $m$ helps

- Columns of $C$ are orthogonal.
- All vectors $y = Ax$ will be distinguishable if we use full $C$.
- If we use less than full $C$, orthogonality is lost, some vectors start to become indistinguishable.
The Role of the Nullspace of $A$

**Theorem**

$2K$-sparse vectors in nullspace of $A \Rightarrow$ exist $K$-sparse $x$ that can’t be recovered by any algorithm.

**Proof.**

Let $z = x_1 - x_2$ be $2K$-sparse $\Rightarrow x_1, x_2$ are $K$-sparse.

$Az = A(x_1 - x_2) = 0 \Rightarrow Ax_1 = Ax_2$
2-D Sparse Polynomial Identification

- What about 2-D polynomials?
- In natural basis: \( f(t, u) = \sum_{i+j=0..Q} \alpha_{ij} t^i u^j \)
- \((t_m, u_m)\) according to some distribution in \((-1, 1) \times (-1, 1)\).

\[
y = \begin{bmatrix} f(t_1, u_1) \\ f(t_2, u_2) \\ \vdots \\ f(t_m, u_m) \end{bmatrix} = \begin{bmatrix} 1 & t_1 & u_1 & t_1 u_1 & t_1^2 & u_1^2 & \cdots \\ 1 & t_2 & u_2 & t_2 u_2 & t_2^2 & u_2^2 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & t_m & u_m & t_m u_m & t_m^2 & u_m^2 & \cdots \end{bmatrix} \begin{bmatrix} \alpha_{00} \\ \alpha_{10} \\ \alpha_{01} \\ \alpha_{11} \\ \alpha_{20} \\ \vdots \end{bmatrix}
\]
Discovering 2D Sparse Polynomials - Recovery Example

\[ x \quad x^* \]

\[
\begin{array}{c|c}
0 & -0.0000 \\
0 & 0.0000 \\
0 & 0.0000 \\
0 & 0.0000 \\
-9.1652 & -9.1652 \\
-3.2926 & -3.2926 \\
0 & -0.0000 \\
0 & -0.0000 \\
\vdots & \vdots \\
0 & 0.0000 \\
0 & -0.0000 \\
2.4264 & 2.4264 \\
0 & -0.0000 \\
0 & -0.0000 \\
\vdots & \vdots \\
0 & 0.0000 \\
0 & -0.0000 \\
4.2605 & 4.2605 \\
0 & 0.0000 \\
\end{array}
\]

\[ \downarrow \text{Recovered from 25 samples on} \]

\[ (0, 1) \times (0, 1) \]
How Well Does It Work?

- Similar to 1-d results.
- Again increasing \( m \) doesn’t change much.
Dimension vs Order

- **1-d:** $f(t) = \sum_{i=0}^{N} x_i t^i$. Goes up to order $N$.
- **2-d:** $f(t) = \sum_{i+j=0}^{Q} x_{ij} t^i u^j$. $N = (Q + 1)(Q + 2)/2$.
- For $Q = 7$, $N = 36$.
- Increasing dimension reduces required order of terms
Did someone say 3-D?

In natural basis: \( f(t, u, v) = \sum_{i+j+k=0..Q} \alpha_{ijk} t^i u^j v^k \)
Discovering 3D Sparse Polynomials - Recovery Example

Recovered from 25 samples on $(0, 1) \times (0, 1) \times (0, 1)$

\[
\begin{array}{cc}
  x & x^* \\
  0 & 0.0000 \\
  0 & -0.0000 \\
 \vdots & \vdots \\
  0 & 0.0000 \\
  0 & -0.0000 \\
 -1.2378 & -1.2378 \\
  0 & 0.0000 \\
  0 & -0.0000 \\
  0 & 0.0000 \\
  0 & 0.0000 \\
  0 & 0.0000 \\
  0 & -0.0000 \\
  6.0409 & 6.0409 \\
  0 & -0.0000 \\
  3.5132 & 3.5132 \\
 -1.3987 & -1.3987 \\
  0 & -0.0000 \\
  0 & -0.0000 \\
 -2.3806 & -2.3806 \\
\end{array}
\]
How Well Does It Work?

- Similar to 2d case.
Summary of Sparse Polynomial Identification

- Can recover very sparse polynomials with a handful of data points.
- In comparison with more ideal bases, Vandermonde has limitations that preclude increasing $K =$ number of non-zero entries.
- For higher dimensions, can avoid use of higher order terms in Vandermonde matrix, but improvement is marginal (if at all).
ODE System Identification: the idea

- Measure an unknown dynamical system at $m$ points in time.
- Determine the system of equations governing the behavior, including parameter values.
- Originally proposed by Wang, Yang, Lai, Kovanis, and Grebogi.
ODE System Identification: the idea

- Measure an unknown dynamical system at \( m \) points in time.
- Determine the system of equations governing the behavior, including parameter values.
- Originally proposed by Wang, Yang, Lai, Kovanis, and Grebogi.
- Similar to polynomial discovery, but \( A \) and \( y \) will come from system measurements.
ODE System Identification - 1d

- Autonomous 1d discrete dynamical map \( x_{n+1} = f(x_n) \)
- Assume \( f \) has form \( f(x) = \sum_{i=0}^{N} \alpha_i x^i \)
- Measure \( f(x_{n_j}), f(x_{n_j+1}), j = 1..m \)
- Try to recover the \( \alpha_i \). Sparse coefficient vector: \( f \) is “simple”.
- Similar to 1d polynomial discovery setup.

\[
y = \begin{bmatrix} x_{j1}+1 \\ x_{j2}+1 \\ \vdots \\ x_{jm}+1 \end{bmatrix} = \begin{bmatrix} 1 & x_{j1} & \cdots & x_{j1}^N \\ 1 & x_{j2} & \cdots & x_{j2}^N \\ \vdots \\ 1 & x_{jm} & \cdots & x_{jm}^N \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_N \end{bmatrix}
\]
Autonomous 1d discrete dynamical map $x_{n+1} = f(x_n)$

Assume $f$ has form $f(x) = \sum_{i=0}^{N} \alpha_i x^i$

Measure $f(x_{nj}), f(x_{nj+1}), j = 1..m$

Try to recover the $\alpha_i$. Sparse coefficient vector: $f$ is “simple”.

Similar to 1d polynomial discovery setup.

Note: $m$ now really means taking $2m$ points, because we must take $x_{nj}$ and $x_{nj+1}$.

Each $(x_i, x_{i+1})$ pair is a single sample in this setup.

$$y = \begin{bmatrix} x_{j_1+1} \\ x_{j_2+1} \\ \vdots \\ x_{j_m+1} \end{bmatrix} = \begin{bmatrix} 1 & x_{j_1} & \cdots & x_{j_1}^N \\ 1 & x_{j_2} & \cdots & x_{j_2}^N \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{j_m} & \cdots & x_{j_m}^N \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_N \end{bmatrix}$$
Example - Logistic Map

- $x_{n+1} = f(x_n) = rx_n(1 - x_n)$
- Coefficient vector: $(0, r, -r, 0, \ldots)$
- We can recover the system equation in chaotic regime taking about 10 sample pairs or more.

Sampling the logistic map, $m = 10$

Recovery error for logistic map, $r = 3.7$
Sensitive to the dynamics determined by $r$.

(Bifurcation diagram: Wikipedia).
How Well Does It Work?

- Right: number of unique system values (to 2 digits).
- When system is too uniform, taking more samples is just more copies of the same data, doesn’t improve the decoding.
- Chaotic regime is best in this case.
In the context of the previous example:

“Parameter Estimation”:
Assume a-priori that $f(x_n) = rx_n - rx_n^2$, then estimate $r$.

“System Identification”:
No a-priori assumption of non-zero terms.
Assume $f(x) = \sum_{i=0}^{N} \alpha_i x^i$
Find non-zero $\alpha_i$ and their values.
Autonomous 2d discrete dynamical map

\[
\begin{bmatrix}
x_{n+1} \\
y_{n+1}
\end{bmatrix} = \begin{bmatrix}
f_1(x_n, y_n) \\
f_2(x_n, y_n)
\end{bmatrix}
\]

Assume \( f_k \) have form

\[
\sum_{i+j=0..Q} \alpha_{ij}^k x^i y^j
\]

Measure system values at \( n_i, i = 1..m \)

Try to recover the \( \alpha_{ij}^k \).

Analogous to the setup of the 2d sparse polynomial discovery, but we repeat the process twice: \( X \to f_1, Y \to f_2 \)

\[
X = \begin{bmatrix}
x_{i_1+1} \\
x_{i_2+1} \\
\vdots \\
x_{i_m+1}
\end{bmatrix} = \begin{bmatrix}
1 & x_{i_1} & y_{i_1} & x_{i_1}y_{i_1} & \cdots \\
1 & x_{i_2} & y_{i_2} & x_{i_2}y_{i_2} & \cdots \\
\vdots & \vdots & \vdots & \vdots & \cdots \\
1 & x_{i_m} & y_{i_m} & x_{i_m}y_{i_m} & \cdots
\end{bmatrix} \begin{bmatrix}
\alpha_{00}^1 \\
\alpha_{10}^1 \\
\alpha_{11}^1 \\
\alpha_{20}^1 \\
\alpha_{02}^1 \\
\vdots
\end{bmatrix}
\]
Analogous to the setup of the 2d sparse polynomial discovery, but we repeat the process twice: $X \rightarrow f_1, Y \rightarrow f_2$

$$X = \begin{bmatrix} x_{i1+1} \\ x_{i2+1} \\ \vdots \\ x_{im+1} \end{bmatrix} = \begin{bmatrix} 1 & x_i & y_i & x_i y_i & \cdots \\ 1 & x_i & y_i & x_i y_i & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_i & y_i & x_i y_i & \cdots \end{bmatrix} \begin{bmatrix} \alpha_{00}^1 \\ \alpha_{10}^1 \\ \alpha_{11}^1 \\ \alpha_{20}^1 \\ \alpha_{02}^1 \end{bmatrix}$$

$$Y = \begin{bmatrix} y_{i1+1} \\ y_{i2+1} \\ \vdots \\ y_{im+1} \end{bmatrix} = \begin{bmatrix} 1 & x_i & y_i & x_i y_i & \cdots \\ 1 & x_i & y_i & x_i y_i & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_i & y_i & x_i y_i & \cdots \end{bmatrix} \begin{bmatrix} \alpha_{00}^2 \\ \alpha_{10}^2 \\ \alpha_{11}^2 \\ \alpha_{20}^2 \\ \alpha_{02}^2 \end{bmatrix}$$
Example - Tinkerbell Map

\[ x_{n+1} = x_n^2 - y_n^2 + ax_n + by_n \]
\[ y_{n+1} = 2x_n y_n + cx_n + dy_n \]

- Coefficient vectors:
  - For x: \((0, a, b, 1, 0, -1, 0, \ldots)\)
  - For y: \((0, c, d, 0, 2, 0, \ldots)\)
How Well Does It Work?

For these system parameters, recovery works well.

With various other parameters not shown, similar results, as long as system is in a stable regime.
Autonomous 3d dynamical system
\[
\frac{df}{(dx,dy,dz)}(x,y,z) = \sum_{i+j+k=0..Q} \alpha_{ijk} x^i y^j z^k
\]

Now a continuous system.

Measure system values and derivatives at \( t_i, i = 1..m \)

Essentially the same problem setup as 3d polynomial discovery.

Perform recovery for each \( X \rightarrow \frac{df}{dx}, Y \rightarrow \frac{df}{dy}, Z \rightarrow \frac{df}{dz} \)

\[
X = \begin{bmatrix}
\frac{df}{dx}(x_1, y_1, z_1) \\
\frac{df}{dx}(x_2, y_2, z_2) \\
\vdots \\
\frac{df}{dx}(x_N, y_N, z_N)
\end{bmatrix} = \begin{bmatrix}
1 & x_1 & y_1 & z_1 & x_1 y_1 & x_1 z_1 & y_1 z_1 & x_1^2 & y_1^2 & \ldots \\
1 & x_2 & y_2 & z_2 & x_2 y_2 & x_2 z_2 & y_2 z_2 & x_2^2 & y_2^2 & \ldots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ldots \\
1 & x_m & y_m & z_m & x_m y_m & x_m z_m & y_m z_m & x_m^2 & y_m^2 & \ldots
\end{bmatrix} \begin{bmatrix}
\alpha_{000} \\
\alpha_{010} \\
\alpha_{001} \\
\alpha_{100} \\
\alpha_{110} \\
\alpha_{101} \\
\vdots \\
\vdots
\end{bmatrix}
\]
Example - Rossler System

\[
\begin{align*}
\frac{dx}{dt} &= -y - z \\
\frac{dy}{dt} &= x + ay \\
\frac{dz}{dt} &= b + z(x - c)
\end{align*}
\]

- Coefficient vectors:
  - For x: \((0, 0, -1, -1, 0, 0, 0, \ldots)\)
  - For y: \((0, 1, a, 0, 0, 0, \ldots)\)
  - For z: \((b, 0, 0, -c, 0, 1, \ldots)\)

- One difference: \(x, y, z\) not in \((-1, 1)\), columns of \(A\) get really bad.

- Normalize each column of \(\Phi\) by \(\sqrt{\sum_{j=1}^{m} \phi_i^2(t_j)}\), rescale \(\alpha\) accordingly after reconstruction.
Example - Rossler System

Sampling the Rossler System, $m = 25$
How Well Does It Work?

\[ ||\alpha^* - \alpha||_2 \]

- Recovery works well here in the chaotic regime: 
  \[ a = 0.1, b = 0.1, c = 18. \]
- Similar results in stable periodic regimes.
- Large number of samples required for \( z \) recovery due to the briefness of excursions away from \( z \approx 0 \) (see figure on previous page).
Summary of ODE System Identification

- Can recover sparse dynamical system equations with a handful of data points.
- Highly stable periodic behaviors may cause issues.
- Number of non-zero terms in power series expansion needs to be low due to Vandermonde basis.
- Can exploit dimension of system to avoid use of higher order terms.
- Noise is an issue that needs to be explored.
PDE System Identification: the idea

- Measure a system at $m$ points in time and space.
- Determine the system of equations governing the behavior, including parameter values.
- Inspired by Prof Platte’s prompt “pode fazer com PDE’s?”
- (in reference to the ODE version of the problem: Wang, Yang, Lai, Kovanis, Grebogi)
PDE Problem Setup

- Will work with 2 spatial dimensions.
- Assume \( u(x, y, t) \) solves a 2nd order PDE and satisfies:

\[
    u_{xx} = \alpha_1 u_t + \alpha_2 u_x + \alpha_3 u_y + \alpha_4 u_{yy} + \alpha_5 u_{xy} + \alpha_6 u_{yt} + \alpha_7 u_{tt}
\]

- Want to measure \( u(x_i, y_i, t_i)_{i=1}^m \) and determine the \( \alpha_j \).
- Will assume instead the ability to measure the derivatives of \( u \).

\[
    y = \begin{bmatrix}
        u_{xx}(x_1, y_1, t_1) \\
        u_{xx}(x_2, y_2, t_2) \\
        \vdots \\
        u_{xx}(x_m, y_m, t_m)
    \end{bmatrix} = \begin{bmatrix}
        \partial(x_1, y_1, t_1) \\
        \partial(x_2, y_2, t_2) \\
        \vdots \\
        \partial(x_m, y_m, t_m)
    \end{bmatrix}
    \begin{bmatrix}
        u_t & u_x & u_y & u_{yy} & u_{xy} & u_{yt} & u_{tt} \\
        u_t & u_x & u_y & u_{yy} & u_{xy} & u_{yt} & u_{tt} \\
        \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
        u_t & u_x & u_y & u_{yy} & u_{xy} & u_{yt} & u_{tt}
    \end{bmatrix}
    \begin{bmatrix}
        \alpha_1 \\
        \alpha_2 \\
        \vdots \\
        \alpha_7
    \end{bmatrix}
\]
Example: Heat Equation

\[ u_{xx} = u_t - u_{yy}, \quad (x, y) \in (0, 1) \times (0, 1) \]
\[ u(0, y, t) = u(1, y, t) = u(x, 0, t) = u(x, 1, t) = 0 \]

Identification of heat system:

\[ \ell_1 \min \]

\[ u_{xx} = u_t - u_{yy} \]
Example: Wave Equation

- \( u_{xx} = u_{tt} - u_{yy}, \ (x, y) \in (0, 1) \times (0, 1) \)
- \( u(0, y, t) = u(1, y, t) = u(x, 0, t) = u(x, 1, t) = 0 \)
- Identification of wave system:

\[
\min_{\ell_1} \quad u_{xx} = u_{tt} - u_{yy}
\]

Recovery error for wave equation with 3 modes in initial conditions
100 experiments per data point
Example: Euler-Tricomi Equation

- $u_{xx} = xu_{yy}$, $(x, y) \in (0, 1) \times (0, 1)$
- No time in this system, so leave time derivatives out of the process.
- Coefficient vector now includes terms of form $x u_x$

$$
\begin{align*}
  u_{xx} &= \alpha_1 u_x + \alpha_2 u_y + \alpha_3 u_{yy} + \alpha_4 u_{xy} \\
          &+ x(\alpha_5 u_x + \alpha_6 u_y + \alpha_7 u_{yy} + \alpha_8 u_{xy}) \\
          &+ y(\alpha_9 u_x + \alpha_{10} u_y + \alpha_{11} u_{yy} + \alpha_{12} u_{xy}) \\
          &+ x^2(\alpha_{13} u_x + \alpha_{14} u_y + \alpha_{15} u_{yy} + \alpha_{16} u_{xy}) \\
          &+ y^2(\alpha_{17} u_x + \alpha_{18} u_y + \alpha_{19} u_{yy} + \alpha_{20} u_{xy})
\end{align*}
$$

- And adjust $A$ accordingly.
Example: Euler-Tricomi Equation

- $u_{xx} = xu_{yy}, \ (x, y) \in (0, 1) \times (0, 1)$
- Identification of ET system:

![Graph showing recovery error for Euler-Tricomi equation with 100 experiments per data point.](image)

```latex
\begin{align*}
\@ (x_i, y_i, t_i)_{i=1}^m \\
\min_{\ell_1} & \\
& u_{xx} = u_{tt} - u_{yy}
\end{align*}
```