Surface Reconstruction from Gradient Fields: **grad2Surf** Version 1.0

Matthew Harker and Paul O'Leary
Institute for Automation
University of Leoben
A-8700 Leoben, Austria
URL: automation.unileoben.ac.at

Original: Aug. 20, 2013

Last Modified: August 20, 2013

Contents

1 Introduction

The Gradient-to-Surface Toolbox, **grad2Surf**, is a MATLAB toolbox for the reconstruction of a surface from its gradient field. The gradient field is assumed to be measured, e.g., via Photometric Stereo, and therefore contains either stochastic or systematic error. It is based on the papers [1], [2] and [3]. The following Table contains a list of the functions and their various purposes.

Function Name	Purpose
g2s	Reconstructs the surface by global least squares (GLS).
g2sSpectral	GLS with Spectral Regularization
g2sTikhonov	GLS with Tikhonov Regularization.
g2sTikhonovStd	Tikhonov Regularization, computes λ
g2sDirichlet	GLS with Dirichlet Boundary Conditions
g2sWeighted	Weighted Least Squares solution
g2sTikhonovRTalpha	Preparatory function for g2sTikhonovRT
g2sTikhonovRT	Real-time surface reconstruction with Tikhonov Regularization
g2sSparse	Solves the GLS problem with traditional sparse methods
g2sSylvester	Solves the necessary Sylvester Equation
g2sTestSurf	Generates a test surface for testing the algorithms

Included with the main functions are auxiliary functions `g2sTestSurf` and `g2sSylvester`, as well as `g2sSparse`, which solves the problem using traditional sparse methods which are common to the literature. It is not recommended to use this function for anything other than comparison purposes.

Note that this package requires the toolbox DOPbox, which is available on MATLAB fileexchange:
`http://www.mathworks.com/matlabcentral/fileexchange/41250`
This document was generated automatically from the file `g2sIntro.m` using the publish2LaTeX package:
`http://www.mathworks.com/matlabcentral/fileexchange/41207`

2 Global Least Squares Solution

Clear the workspace, and change some of the graphics settings.

```
close all
clear all
setUpGraphics(8) ;
```

Set the (discrete) size of the surface, and then generate the test surface:

```
m = 156 ;
n = 213 ;
figure ;
[Ztrue,Zx,Zy,x,y] = g2sTestSurf(m,n,'even',1) ;
```

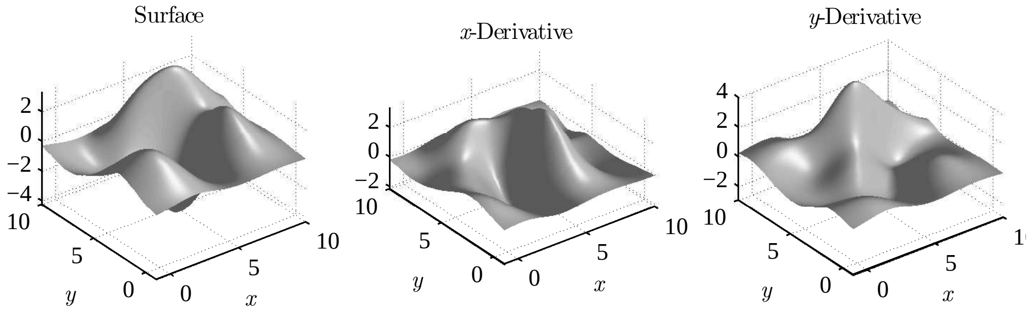

Figure 1: The test surface and its gradient field.

Add noise to the gradient field:

```
sigma = 0.05 ;
Ax = ( max( Zx(:) ) - min( Zx(:) ) )/2 ;
Ay = ( max( Zy(:) ) - min( Zy(:) ) )/2 ;
ZxN = Zx + sigma * Ax * randn(m,n) ;
ZyN = Zy + sigma * Ay * randn(m,n) ;
figure
subplot(1,2,1) ;
imagesc( x, y, ZxN ) ;
axis equal, axis tight
subplot(1,2,2) ;
imagesc( x, y, ZyN ) ;
axis equal, axis tight
colormap(gray) ;
```

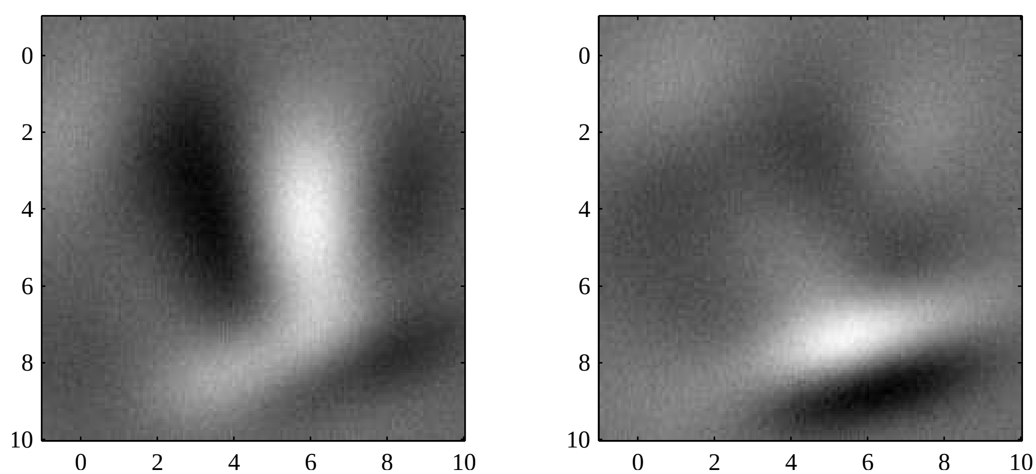

Figure 2: Noisy gradient field: Gaussian noise with a standard deviation that is 5% of the gradient's amplitude.

Set the number of points to be used in the numerical derivatives, `N`, and test the basic least squares solution:

```
N = 3 ;
```

```
tic ;
Z = g2s( ZxN, ZyN, x, y, N ) ;
tGLS = toc
h1 = figure ;
subplot(1,2,1) ;
g2sPlotSurf( x, y, Z, h1, 'Surface Reconstructed with GLS' ) ;
%
tic ;
warning('off','WarnTests:convertTest') ;
% Note: "g2sSparse" is for comparison purposes only, and contains
% a warning that you should use "g2s" instead
Zsparse = g2sSparse( ZxN, ZyN, x, y, N ) ;
warning('on','WarnTests:convertTest') ;
tSparse = toc
subplot(1,2,2) ;
g2sPlotSurf( x, y, Zsparse, h1, 'Reconstruction with Sparse Method' ) ;
```

```
tGLS =

    0.2436


tSparse =

    1.2911
```

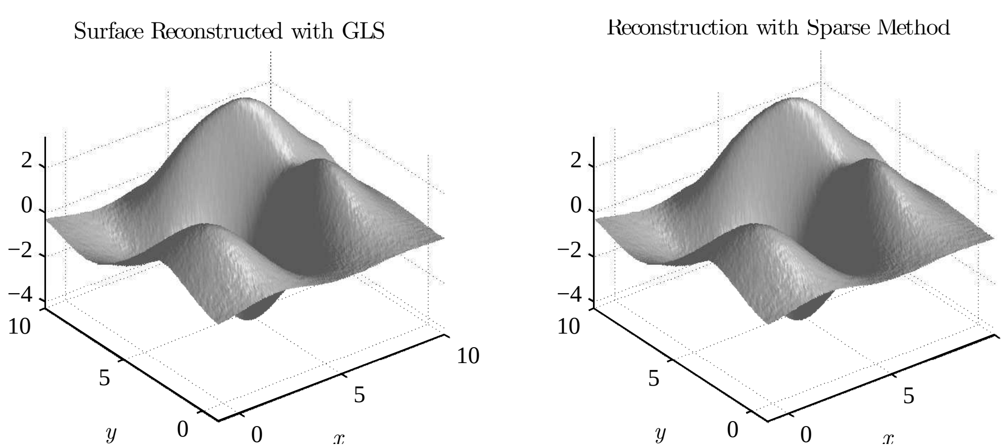

Figure 3: The sparse method solves the GLS problem, however, by vectorizing the system of equations. It is therefore far less efficient (an $O(n^6)$ algorithm instead of an $O(n^3)$ algorithm) and less accurate, since using sparse methods only an approximate solution is obtained.

3 Spectral Regularization, Constrained Regularization, Weighte Least Squares

For spectral regularization the surface is described by a set of orthogonal basis functions. The regularizing effect is by truncating the set of basis functions by any form of low-, high-, or bandpass-filter. Note that if the full set of basis functions is used (no-filter) the solution is identical to the GLS solution.

```
basisFns = 'cosine' ;
p = floor(m/4) ;
q = floor(n/4) ;
Mask = [p,q] ;
% The following (comment) produces a band pass mask:
%Mask = zeros(m,n) ;
%Mask(1:p,1:q) = ones(p,q) ;
%Mask(1:2,1:2) = zeros(2) ;
Zpoly = g2sSpectral( ZxN, ZyN, x, y, N, Mask, basisFns ) ;
h3a = figure ;
g2sPlotSurf( x, y, Zpoly, h3a, 'Reconstructed Surface (Spectral)' ) ;
```

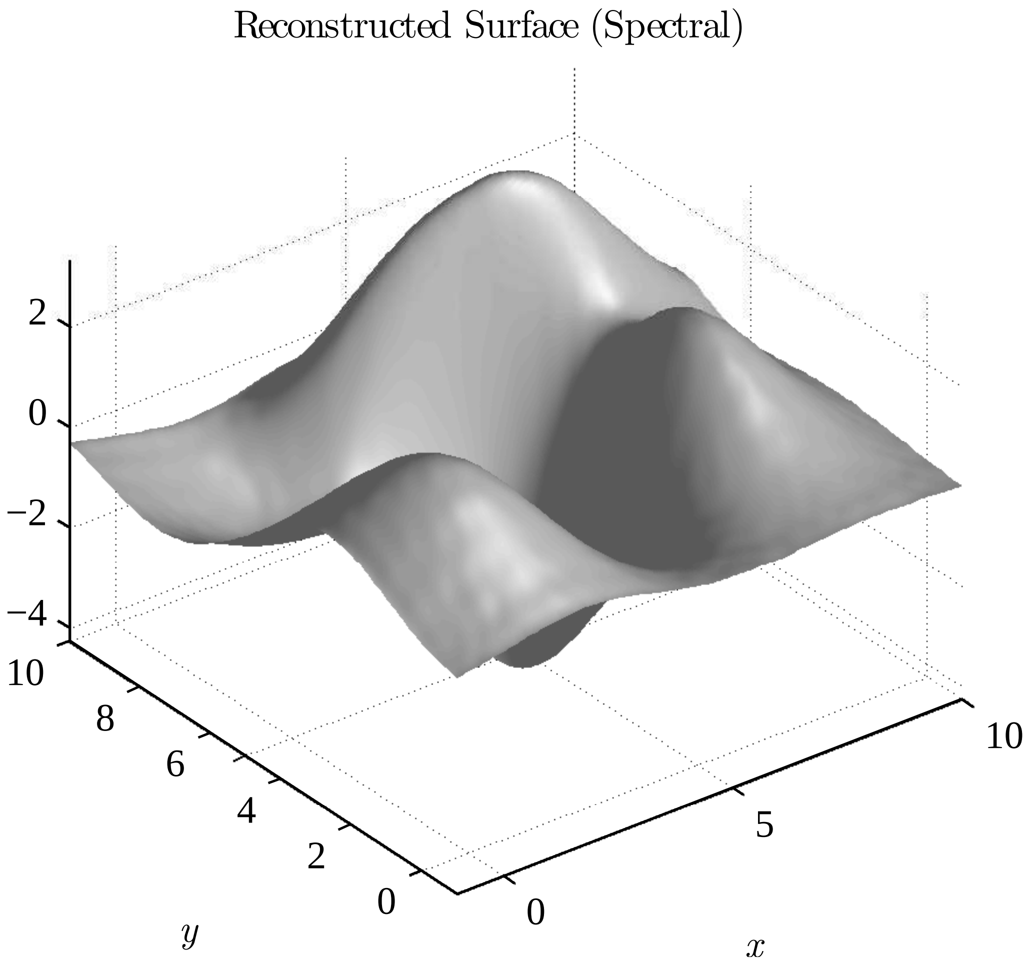

Figure 4: The surface reconstruction using cosine basis functions for spectral regularization. Only one quarter of the low-frequency basis functions are used, effecting a low pass filter which smooths the reconstructed surface. The computation is also more efficient than the full solution.

Dirichlet Boundary Conditions specify the value of the integral surface over the boundary of the domain. This can be used for a regularizing effect, e.g., by setting the boundary values to zero. It is particularly effective and robust to noise if the boundary values are known a-priori.

```
ZB = zeros(m,n) ;
ZB(1,:) = sin(2*x)' ; ZB(m,:) = sin(2*x) ;
ZB(:,1) = sin(2*y) ; ZB(:,n) = sin(2*y) ;
Zdir = g2sDirichlet( Zx, Zy, x, y, 5, ZB ) ;
figure
g2sPlotSurf( x, y, Zdir ) ;
```

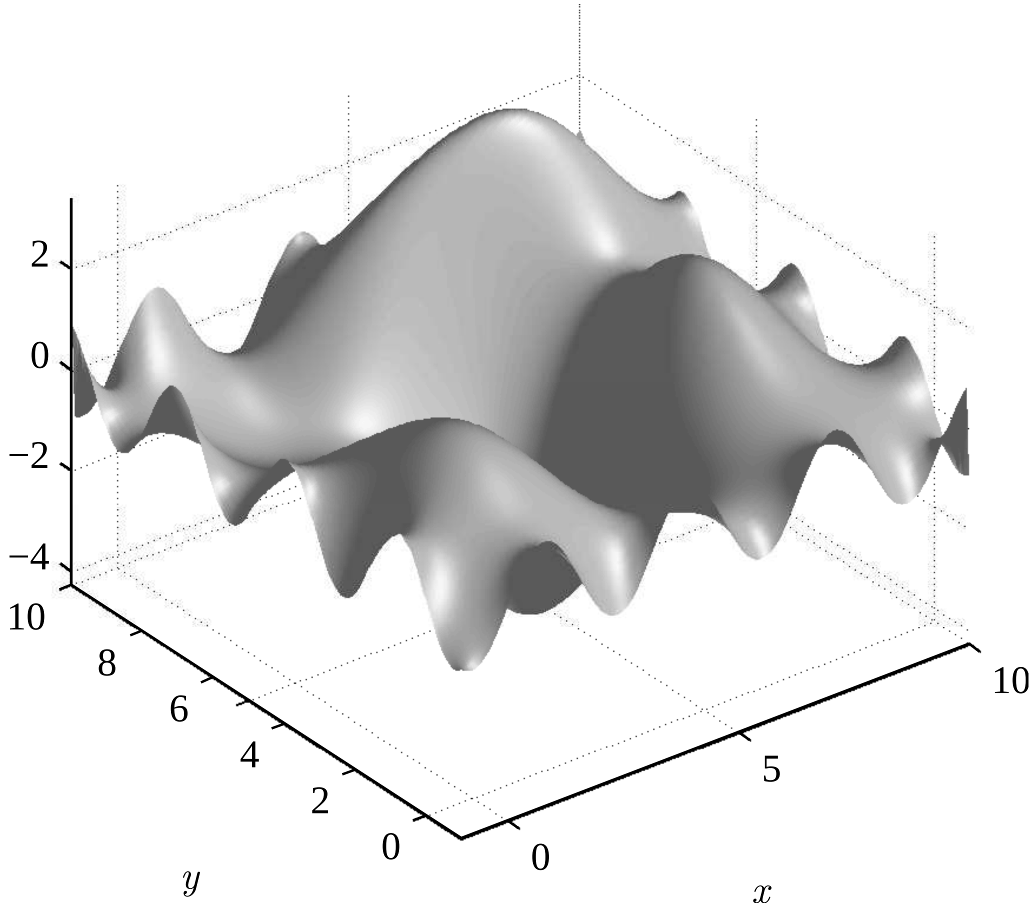

Figure 5: For Dirichlet boundary conditions, the value of the surface at the boundary can be specified arbitrarily, as in this example.

Weighted Least Squares is used when the gradient field is corrupted by heteroscedastic Gaussian noise. This occurs particularly in Photometric Stereo, since one of the main assumptions is that the camera effects an orthogaphic projection. Since this is not the case, the WLS squares can be used to compensate for this.

```
f = 1 + exp( -(x-mean(x)).^2/(2*std(x)^2) ) ; f = f/mean(f) ;
g = 1 + exp( -(y-mean(y)).^2/(2*std(y)^2) ) ; g = g/mean(g) ;
Lxx = diag( f ) ; Lxy = diag( g ) ;
Lyx = diag( f ) ; Lyy = diag( g ) ;
ZxNw = Zx + sigma * Ax * sqrtm(Lxy)*randn(m,n)*sqrtm(Lxx) ;
ZyNw = Zy + sigma * Ay * sqrtm(Lyy)*randn(m,n)*sqrtm(Lyx) ;
ZwLS = g2s( ZxNw, ZyNw, x, y, N ) ;
Zw = g2sWeighted( ZxNw, ZyNw, x, y, N, Lxx, Lxy, Lyx, Lyy ) ;
h5 = figure ;
subplot(1,3,1) ;
g2sPlotSurf( x, y, Ztrue, h5, 'Exact' ) ;
subplot(1,3,2) ;
g2sPlotSurf( x, y, ZwLS, h5, 'GLS' ) ;
subplot(1,3,3) ;
```

```
g2sPlotSurf( x, y, Zw, h5, 'Weighted LS' ) ;
```

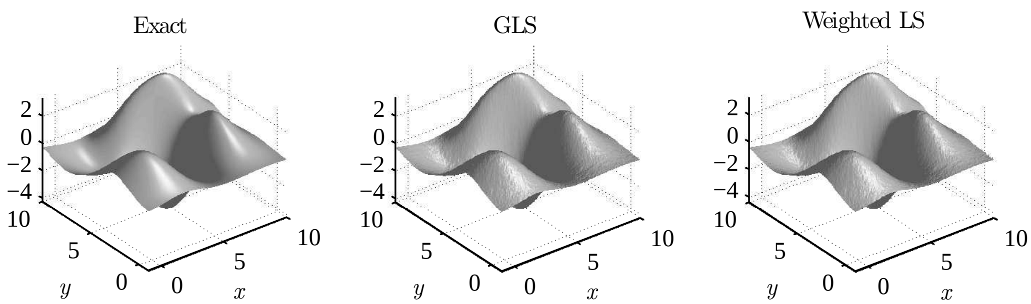

Figure 6: The true surface, and reconstructions from gradient fields subject to heteroscedastic Gaussian noise. In the case the Weighted Least Squares solution is optimal in the Maximum Likelihood sense.

4 Tikhonov Regularization

For Tikhonov regularization, if the value of λ is known, the function `g2sTikhonov` can be used, which implements the GLS solution with Tikhonov regularization of arbitrary degree. Tikhonov regularization in "Standard form" is of degree zero, and the a-priori estimate of the surface is `Z0=zeros(m,n)`, i.e., a flat surface.

```
lambda = 0.025 ;
deg = 0 ;
Z0 = zeros(m,n) ;
[ Ztik, Res ] = g2sTikhonov( ZxN, ZyN, x, y, N, lambda, deg, Z0 ) ;
h3 = figure ;
g2sPlotSurf( x, y, Ztik, h3, 'GLS Solution with Tikhonov Regularization'
            ) ;
```

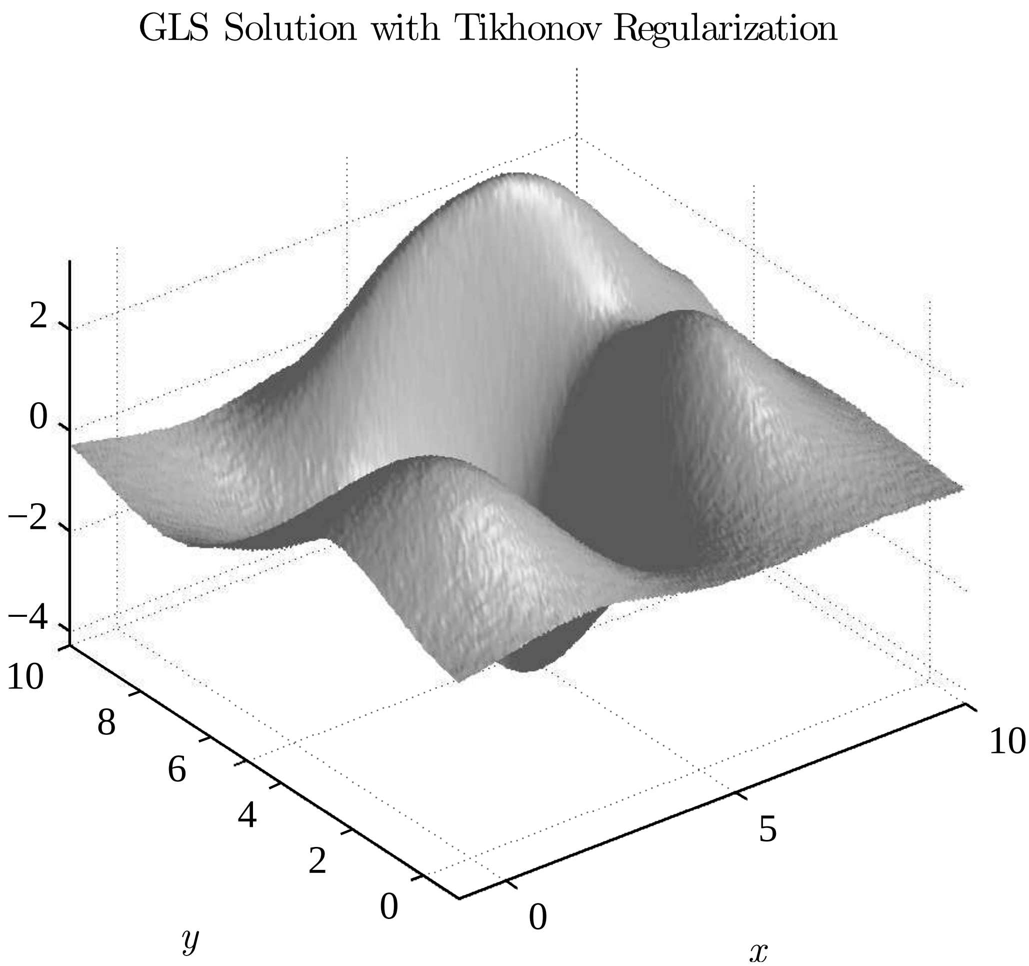

Figure 7: Tikihonov regularization adds a penalty term to the GLS cost funtion. Depending on the degree of the penalty term, the deviation of the reconstructed surface from the a-priori estimate is penalized, or its first or second derivative equivalents.

To compute a value for λ, use the L-Curve

```
noLambdas = 100 ;
[ ZtikL, lamOpt, RC, Theta ] = g2sTikhonovStd( Zx, Zy, x, y, N,
          noLambdas, Z0 ) ;
%
figure
loglog( RC(:,2), RC(:,3), 'r.-' ) ;
hold on
text( RC(1,2), RC(1,3), ['lambda = ', num2str(RC(1,1))] ) ;
text( RC(noLambdas,2), RC(noLambdas,3), ['lambda = ', num2str(RC(
          noLambdas,1))] ) ;
title('L-Curve') ;
```

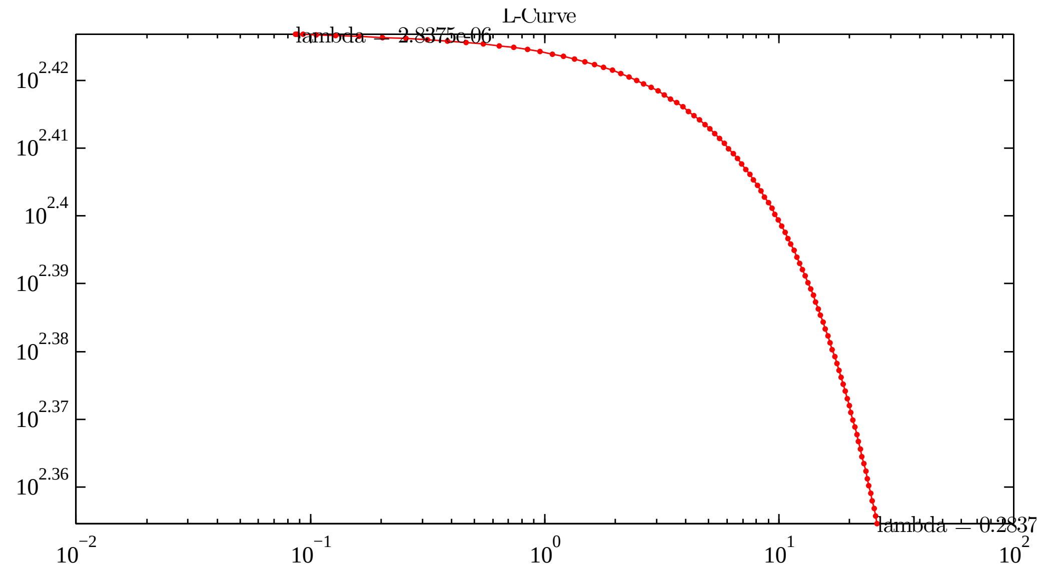

Figure 8: The L-Curve is only one method for determining the regularization parameter for Tikhonov Regularization.

Finally, for Real-Time implementations of Tikhonov, an algorithm has been developed [3] which separates the computation into everything which can be computed beforehand (without the measured gradient field), and the computation which needs to be done in a Real-Time environment. The function `g2sTikhonovRTalpha` makes the preparatory computations based solely on the domain information and the degree of accuracy required. The cell array `S` is then passed on to the Real-Time portions of the code, the function `g2sTikhonovRT`. For this example, with a 156×213 surface, the reconstrution time is about 1/10th of the preparatory computation. The relative error between the full algorithm an the real time algorithm is negligibly small.

```
tic ;
S = g2sTikhonovRTalpha( x, y, N ) ;
tPrep = toc
%
tic
ZtikRT = g2sTikhonovRT( ZxN, ZyN, S, lambda, Z0 ) ;
tRT = toc
%
eRel = 100 * norm( ZtikRT - Ztik, 'fro' ) / norm( Ztik, 'fro' )
```

```
tPrep =

    0.0797


tRT =

    0.0057
```

```
eRel =

   1.7471e-08
```

References

[1] M. Harker and P. O'Leary. Least squares surface reconstruction from measured gradient fields. In *CVPR 2008*, pages 1–7, Anchorage, AK, 2008. IEEE.

[2] M. Harker and P. O'Leary. Least squares surface reconstruction from gradients: Direct algebraic methods with spectral, Tikhonov, and constrained regularization. In *IEEE CVPR*, pages 2529–2536, Colorado Springs, CO, 2011. IEEE.

[3] Matthew Harker and Paul OLeary. Direct regularized surface reconstruction from gradients for industrial photometric stereo. *Computers in Industry*, (0):–, 2013.