

# User Guide for **deconvtv** (MATLAB Version 1.0)

Stanley H. Chan\*, Philip E. Gill† and Truong Q. Nguyen ‡

September 23, 2013

## 1 Introduction

**deconvtv** is a numerical algorithm for solving total variation constrained least-squares problems. The concept of the algorithm is based on an augmented Lagrangian method proposed in [1], and is a variation of the popularly known Alternating Direction Methods of Multipliers (ADMM) [2, 3]. In particular, **deconvtv** solves the following minimization problems

$$\underset{\mathbf{f}}{\text{minimize}} \quad \frac{\mu}{2} \|\mathbf{H}\mathbf{f} - \mathbf{g}\|_2^2 + \|\mathbf{f}\|_{TV}, \quad (1)$$

$$\underset{\mathbf{f}}{\text{minimize}} \quad \mu \|\mathbf{H}\mathbf{f} - \mathbf{g}\|_1 + \|\mathbf{f}\|_{TV}, \quad (2)$$

where  $\mathbf{H}$  is a circulant matrix denoting a spatially invariant linear operator,  $\mu$  is a regularization parameter, and  $\|\mathbf{f}\|_{TV}$  is the total variation norm of the data  $\mathbf{f}$ , defined as

$$\|\mathbf{f}\|_{TV} = \sum_k \sqrt{\beta_x [\mathbf{D}_x \mathbf{f}]_k^2 + \beta_y [\mathbf{D}_y \mathbf{f}]_k^2 + \beta_t [\mathbf{D}_t \mathbf{f}]_k^2}.$$

Here,  $[\mathbf{f}]_k$  is the  $k$ -th entry of the vector  $\mathbf{f}$ . The operators  $\mathbf{D}_x$ ,  $\mathbf{D}_y$  and  $\mathbf{D}_t$  are the gradient operators along the horizontal, vertical and temporal directions. The relative emphasis of  $\mathbf{D}_x$ ,  $\mathbf{D}_y$  and  $\mathbf{D}_t$  can be controlled by  $\beta_x$ ,  $\beta_y$  and  $\beta_t$ , respectively.

## 2 User Interface

The user interface of **deconvtv** is as follows:

```
out = deconvtv(g, h, mu, opts);
```

### 2.1 Input Variables

- **g**: Input image. It can be a gray-scaled image, color image, or gray-scaled video.
- **h**: Point spread function, could be a two-dimensional matrix, or a three-dimensional tensor.
- **mu**: Regularization parameter  $\mu$ .
- **opts**: A structure of options (See below).

---

\*School of Engineering and Applied Sciences, Harvard University.

†Department of Mathematics, University of California, San Diego.

‡Department of Electrical and Computer Engineering, University of California, San Diego.

## 2.2 Option Fields

All default settings are marked in  $\{\cdot\}$ .

- `opts.method`: Defines the method to be used, either '11' or {'12'}. If '12' is chosen, then `deconvtv` solves Problem (1). If '11' is chosen, then `deconvtv` solves Problem (2).
- `opts.beta`: An  $1 \times 3$  vector specifying  $(\beta_x, \beta_y, \beta_t)$ . Default is  $\{[1, 1, 0]\}$ . For video deblurring and denoising, `opts.beta` can be chosen as  $[1, 1, 2.5]$ .
- `opts.rho_r`: Regularization parameter to the constraint violation  $\|\mathbf{u} - \mathbf{Df}\|^2$  (See [1]). Default is  $\{2\}$ .
- `opts.rho_o`: Regularization parameter to the constraint violation  $\|\mathbf{Hf} - \mathbf{g} - \mathbf{r}\|^2$  (See [1]). Default is  $\{50\}$ .
- `opts.alpha`: Criteria for `rho_r` update (See [1]). Default is  $\{0.7\}$ .
- `opts.gamma`: Update constant for `rho_r` (See [1]). Default is  $\{2\}$ .
- `opts.max_itr`: Maximum number of iteration  $\{20\}$ .
- `opts.tol`: Tolerance level of relative change. Default is  $\{1e-3\}$ .
- `opts.print`: Print intermediate report, either 'true' or 'false'. Default is  $\{\text{false}\}$ .
- `opts.f`: Initial guess. Default is  $\{\mathbf{g}\}$ .
- `opts.y1`, `opts.y2`, `opts.y3`: Initial guess of Lagrange multipliers for the constraint  $\mathbf{u} = \mathbf{Df}$ . Default is  $\{\mathbf{0}\}$ .
- `opts.z`: Initial guess of Lagrange multiplier for the constraint  $\mathbf{r} = \mathbf{Hf} - \mathbf{g}$ . Default is  $\{\mathbf{0}\}$ .

## 2.3 Output Variables

- `out.f`: Output image, or video.
- `out_itr`: total number of iterations elapsed
- `out.relchg`: final relative change
- `out.Df1`, `out.Df2`, `out.Df3`: Output image gradients
- `out.y1`, `out.y2`, `out.y3`: Output Lagrange multipliers
- `out.rho_r`: final regularization parameter

## 3 Examples

### 3.1 Image Denoising

#### Image Denoising Example

```
% Prepare images
f_orig = im2double(imread('./data/wind.jpg'));
[rows cols frames] = size(f_orig);
H       = fspecial('gaussian', [9 9], 2);
g       = imfilter(f_orig, H, 'circular');
g       = imnoise(g, 'salt & pepper', 0.05);

% Setup parameters (for example)
opts.rho_r = 5;           opts.rho_o = 100;           opts.beta = [1 1 0];
opts.print = true;        opts.alpha = 0.7;           opts.method = 'l1';

% Setup mu
mu = 20;

% Main routine
tic
out = deconvtv(g, H, mu, opts);
toc

% Display results
figure(1); imshow(g);      title('input');
figure(2); imshow(out.f);  title('recovered');
```



(a) Input Image



(b) Recovered Image

Figure 1: Example: Image denoising. Time elapsed: 2.2 seconds.

### 3.2 Image Deblurring

#### Image Deblurring Example

```
% Prepare images
f_orig = im2double(imread('./data/building.jpg'));
[rows cols frames] = size(f_orig);
H       = fspecial('gaussian', [9 9], 2);
g       = imfilter(f_orig, H, 'circular');
g       = imnoise(g, 'gaussian', 0, 0.00001);

% Setup parameters (for example)
opts.rho_r = 2;      opts.beta  = [1 1 0];    opts.print = true;
opts.alpha = 0.7;    opts.method = 'l2';

% Setup mu
mu = 10000;

% Main routine
tic
out = deconvtv(g, H, mu, opts);
toc

% Display results
figure(1); imshow(g);      title('input');
figure(2); imshow(out.f);  title('recovered');
```



(a) Input Image



(b) Recovered Image

Figure 2: Example: Image deblurring. Time elapsed: 8.24 seconds.

### 3.3 Video Disparity Refinement

Video Disparity Refinement Example

```
folder_name = './data/';

fname = sprintf('%sdata%04d.jpg', folder_name, 1);
f = im2double(imread(fname));
[rows cols frames] = size(f);
g = zeros(rows,cols,frames);

for fidx = 1:10
    fname = sprintf('%sdata%04d.jpg', folder_name, fidx);
    f = im2double(imread(fname));
    if size(f,3)>1
        g(:,:,fidx) = rgb2gray(f);
    else
        g(:,:,fidx) = f;
    end
end

% Setup parameters (for example)
opts.beta = [1 1 10];    opts.print = true;    opts.method = 'l1';

% Setup mu
mu = 1;

% Main routine
tic
out = deconvtv(g, 1, mu, opts);
toc

% Display results
figure(1); imshow(g(:,:,5));    title('input');
figure(2); imshow(out.f(:,:,5)); title('recovered');
```



Figure 3: Example: Video disparity refinement. Top: Input. Bottom: Recovered. Time elapsed: 12.1 seconds for 10 frames.

## 4 Copyright

This software is Copyright ©2011 The Regents of the University of California. All Rights Reserved.

Permission to use, copy, modify, and distribute this software and its documentation for educational, research and non-profit purposes, without fee, and without a written agreement is hereby granted, provided that the above copyright notice, this paragraph and the following three paragraphs appear in all copies.

Permission to make commercial use of this software may be obtained by contacting: Technology Transfer Office 9500 Gilman Drive, Mail Code 0910 University of California La Jolla, CA 92093-0910 (858) 534-5815 invent@ucsd.edu

This software program and documentation are copyrighted by The Regents of the University of California. The software program and documentation are supplied “as is”, without any accompanying services from The Regents. The Regents does not warrant that the operation of the program will be uninterrupted or error-free. The end-user understands that the program was developed for research purposes and is advised not to rely exclusively on the program for any reason.

IN NO EVENT SHALL THE UNIVERSITY OF CALIFORNIA BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, INCLUDING LOST PROFITS, ARISING OUT OF THE USE OF THIS SOFTWARE AND ITS DOCUMENTATION, EVEN IF THE UNIVERSITY OF CALIFORNIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. THE UNIVERSITY OF CALIFORNIA SPECIFICALLY DISCLAIMS ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE SOFTWARE PROVIDED HEREUNDER IS ON AN “AS IS” BASIS, AND THE UNIVERSITY OF CALIFORNIA HAS NO OBLIGATIONS TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS.

## References

- [1] S.H. Chan, R. Khoshabeh, K.B. Gibson, P.E. Gill, and T.Q. Nguyen, “An augmented Lagrangian method for total variation image restoration,” *IEEE Trans. Image Process.*, vol. 20, no. 11, pp. 3097–3111, Nov. 2011.
- [2] Y. Wang, J. Yang, W. Yin, and Y. Zhang, “A new alternating minimization algorithm for total variation image reconstruction,” *SIAM Journal on Imaging Sciences*, vol. 1, pp. 248–272, 2008.
- [3] B. Wahlberg, S. Boyd, M. Annergren, and Y. Wang, “An ADMM algorithm for a class of total variation regularized estimation problems,” in *Proceedings 16th IFAC Symposium on System Identification*, Jul. 2012, vol. 16.