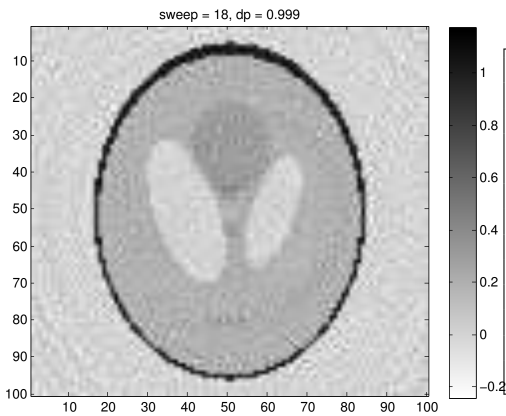
Codes for each problem are available at `https://github.com/kjoyce/inverse_problems/tree/master/homework06`

1. Bardsley 3.14. Modify `Tomography.m` so that it implements Kaczmarz's Method. How many sweeps through all of the indicies, i.e. implementations of the method, does it take to obtain a good reconstruction?

Solution:

Adding the following lines of code implements the method. We ues the DP stopping criterion to determine the amount of iterations.




```
x = A(1,:)';
i = 1;
while 1
  i = i + 1;
  x  = x + (b(i) - A(i,:)*x)/norm(A(i,:))^2 * A(i,:)';
  dp = norm( A*x - b(:) )/(size(A,1)*noise^2);
  if i == size(A,1)
    i = 0;
  end
  if dp <= 1
    break
  end
end
```

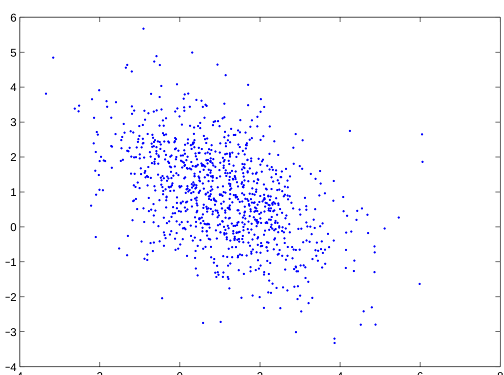
□

2. Bardsley 4.3. Sampling from Gaussian probability densities.

(a) Let $\boldsymbol{B}$ be an $m \times n$ matrix, $\boldsymbol{v} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{C})$, and $\boldsymbol{w} = \boldsymbol{B}\boldsymbol{v}$, then $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{B}\boldsymbol{\mu}, \boldsymbol{B}\boldsymbol{C}\boldsymbol{B}^T)$.† Use this to show that if $\boldsymbol{R}$ is a square root matrix for $\boldsymbol{C}$, i.e. $\boldsymbol{C} = \boldsymbol{R}^T\boldsymbol{R}$, and if $\boldsymbol{v} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ then $\boldsymbol{w} = \boldsymbol{\mu} + \boldsymbol{R}^T\boldsymbol{v}$ has distribution $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{C})$.

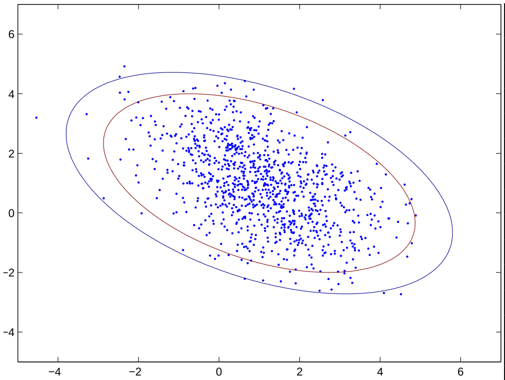
By linearity, $E[\boldsymbol{w}] = \boldsymbol{\mu} + \boldsymbol{R}^T\boldsymbol{0} = \boldsymbol{\mu}$, and by †, $Var[\boldsymbol{w}] = \boldsymbol{R}^T\boldsymbol{I}\boldsymbol{R} = \boldsymbol{C}$.

(b) Use MATLAB and part (a) to compute and plot 1000 samples from the random vector $\boldsymbol{w} \sim \mathcal{N}\left(\begin{bmatrix}1\\1\end{bmatrix}, \begin{bmatrix}2 & -1\\-1 & 2\end{bmatrix}\right)$.



```
n = 1000
v = randn(2,n);
mu = ones(2,n);
C = [2 -1;-1 2];
R = chol(C);
w = mu + R'*v;
plot(w(1,:),w(2,:),'.') ,xlim([-5,7]) ,ylim([-5,7]);
```

(c) If $\boldsymbol{v} = (v_1, \dots, v_k)$ is a random vector such that $v_i \sim \mathcal{N}(0,1)$ for $i = 1, \dots, k$, then $Q = \sum_{i=1}^k w_i^2$ is a $\chi^2(k)$ random variable. Verify that, approximately, a correct amount of the sampled points from part (b) are located inside the confidence regions given by 95% and 99% limits of the $\chi^2(2)$ distribution, computed using the `chi2inv` funciton. Note that you need to normalize the samples, i.e. look at $\boldsymbol{C}^{-1/2}(\boldsymbol{\mu} - \boldsymbol{w}_i)$.




```
vv = R\(w - mu);
xi = sum(vv.^2);
alpha = .05;
l1 = chi2inv((1-alpha),2);
alpha = .01;
l2 = chi2inv((1-alpha),2);
sum(xi <= l1)/length(xi)
ans =

    0.9490

sum(xi <= l2)/length(xi)
ans =

    0.9920
```

3. Bardsley 4.4. GMRF with periodic boundary conditions.

Modify `GMRFDirichlet.m` so that it computes samples from (4.16) where $\boldsymbol{L}_{1D}$ is given by (4.17) in one dimension and (4.14) in two dimensions. Note that these precision matrices have a zero eigenvalue, so you cannot use the Cholesky factorization. Instead, Compute the square root of $\boldsymbol{L}$ and $\boldsymbol{L}^\dagger$ using an eigenvalue decomposition. To compute the eigen values decomposition in two dimensions, note that multiplication by $\boldsymbol{L}_{2D}$ is equivalent to discrete convolution with the $n \times n$ kernel

$$\boldsymbol{l} = \begin{bmatrix} l_{-n/2,n/2-1} & \cdots & l_{0,n/2-1} & \cdots & l_{n/2-1,n/2-1} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ l_{-n/2,0} & \cdots & l_{0,0} & \cdots & l_{n/2-1,0} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ l_{-n/2,-n/2} & \cdots & l_{0,-n/2} & \cdots & l_{n/2-1,-n/2} \end{bmatrix},$$

where $l_{0,0} = 4$, $l_{-1,0} = l_{1,0} = l_{0,-1} = l_{0,1} = -1$ and $l_{ij} = 0$ otherwise. The periodic boundary condition makes $\boldsymbol{L}_{2D}$ a BCCB matrix with eigenvalues $\widehat{\boldsymbol{l}}_{\boldsymbol{s}} = n\mathrm{DFT}(\boldsymbol{l}_s)$, where $\boldsymbol{l}_s = \texttt{fftshift}(\boldsymbol{l})$, and that $\boldsymbol{L}\boldsymbol{x} = \mathrm{vec}\Big(\mathrm{IDFT}(\widehat{\boldsymbol{l}}_{\boldsymbol{s}} \odot \mathrm{DFT}(\boldsymbol{X})\Big)$.

Use this factorization, and the corresponding factorization for $\boldsymbol{L}^\dagger$, to compute samples in the two dimensional case.