Since there are $4m$ Lagrange multipliers $(\lambda, \mu, \alpha, \beta)$, we need to pad the $2m \times 2m$ matrix P with zeros to make it into a $4m \times 4m$ matrix

$$P_{2a} = \begin{pmatrix} P & 0_{2m,2m} \\ 0_{2m,2m} & 0_{2m,2m} \end{pmatrix}.$$

Similarly, we pad q with zeros to make it a vector q_{2a} of dimension $4m$,

$$q_{2a} = \begin{pmatrix} q \\ 0_{2m} \end{pmatrix}.$$

We implemented the above methods in `Matlab`; see Appendix B, Section B.4. Choosing $C = m$ is typically a good choice because then the values of λ_i and μ_j are not too small $(C/m = 1)$. If C is chosen too small, we found that numerical instability increases drastically and very poor results are obtained. Increasing C tends to encourage sparsity.

We ran our `Matlab` implementation of the above method on the set of 50 points generated at random by the program shown below with $C = 50$ and various values of ν starting with $\nu = 0.03$:

```
X13 = 15*randn(50,1);
ww13 = 1;
y13 = X13*ww13 + 10*randn(50,1) + 20;
[~,~,~,~,~,~,~,~,w1] = runuregb(rho,0.03,X13,y13,50)
```

Figure 56.11 shows the result of running the program with $\nu = 0.03$. We have $p_f = 0, q_f = 0, p_m = 2$ and $q_m = 1$. There are 47 points strictly inside the slab. The slab is large enough to contain all the data points, so none of them is considered an error.

The next value of ν is $\nu = 0.21$, see Figure 56.12. We have $p_f = 4, q_f = 5, p_m = 6$ and $q_m = 6$. There are 38 points strictly inside the slab.