```
$initializeu_0$;
$r_0 = b - Au_0$;
$L2normr0 = L2norm(r_0)$;
$p_0 = r_0$;
$niter = 0$;
while ($niter < nitermax$) do
    $niter = niter + 1$;
    $alpha = (r_n^T r_n)/(p_n^T A p_n)$;
    $u_{n+1} = u_n + alpha_n p_n$;
    $r_{n+1} = r_n - alpha_n A p_n$;
    if ($L2normr/L2norm0 < threshold$) then
        break;
    end
    $beta_n = (r_{n+1}^T rn + 1)/(r_n^T r_n)$;
    $p_{n+1} = rn + 1 + beta_n p_n$;
end
```

Algorithm 1: Conjugate Gradient pseudo-code

In the implementation of CGSolver, I used 5 different functions which I defined in matvecops.cpp
L2norm − This was used to calculate the L2-norm of a vector.
dotProduct − This was used to calculate the dot product of two vectors.
matVecProduct − This was used to calculate the matrix vector product of a CSR matrix and a vector.
scalVecProduct − This was used to calculate the product of a vector and a scalar.
sum2Vec − This was used to get the sum of two vectors.

The use of these five functions greatly reduced the length of my code and made debugging easier.