We often see linear systems (Ax=b), which are
- **sparse**, 
- **symmetric**, and
- **positive definite**.

An example is the linearized implicit update step.

We could solve this with Gaussian Elimination, for example, but a much better idea is the Conjugate Gradient method, which only requires matrix multiplies.

The ordinary dot product
\(<x, y> = x^T y\)

is symmetric, positive definite, and linear in both its arguments.

Using this generalized dot product (\(<x, y>_A\)), we can create a basis \(P\) which is orthonormal under the dot product:

\[ P^TAP = I \]

but, recall that we wanted to solve the following equation:

\[ Ax = b \]

the first step is to multiply (2) by \(P^T\)

\[ P^TAx = P^Tb \]

the create a new variable \(y\) such that

\[ x = Py \]

Substituting (1) and (4) into (3) we get:

\[ P^TAPy = y = P^Tb \]

which allows us to solve for \(x\)

\[ x = Py = PP^Tb \]

The key to the conjugate gradient method is to build up a sequence of bases \(P_1, P_2, \ldots\) one vector at a time (orthonormalizing each new vector to ensure \(P_i^TAP_i = I\)), and solve for successive approximations to \(x_1, x_2, \ldots \rightarrow x\) by using equation (6): \(x_i = P_iP_i^Tb\).