One of the most wonderful and surprising connections in the field of matrix computations is the elegant interplay between matrices and orthogonal polynomials. These relationships lead to magnificently simple insights into complicated methods. In this lecture we shall unshroud some of these connections. However, the field is so deep. Probably the best textbook on the topic was written by Purdue's own Walter Gautschi. It is tersely titled: "Orthogonal Polynomials."

1 WHAT ARE ORTHOGONAL POLYNOMIALS?

For the purposes of this lecture, a polynomial is a univariate function:

$$p(t) = \sum_{i=0}^{n} p_i t^i.$$  

For example:

$$p(t) = \frac{1}{2} t^2 - \frac{1}{2}$$

$$q(t) = \frac{5}{2} t^3 - \frac{3}{2} t.$$  

The degree of a polynomial is the power of the largest term. In the generic polynomial definition, the degree is $n$. For the examples of $p$ and $q$, the degrees are 2 and 3 respectively. A polynomial of degree 0 is a constant.

We call two polynomials orthogonal if:

$$\int_{-1}^{1} p(t) q(t) \, dt = 0.$$  

This is a type of continuous analog of two vectors:

$$v^T u = \sum_{i=1}^{n} v_i u_i = 0.$$  

EXEMPLARY 1  The two polynomials $p$ and $q$ given above are orthogonal.

$$\int_{-1}^{1} p(t) q(t) \, dt = \int_{-1}^{1} \frac{1}{2} (t^2 - 1) \frac{1}{2} (5t^3 - 3t) \, dt$$

$$= \frac{1}{4} \int_{-1}^{1} 5t^5 - 5t^3 - 3t^3 + 3t \, dt$$

$$= \frac{1}{4} \int_{-1}^{1} 5t^5 - 8t^3 + 3t \, dt.$$  

All of these terms are odd functions, and they are integrated over a symmetric region, hence the result is 0.

More generally, we can consider polynomials that are orthogonal with respect to

an arbitrary interval  \( \int_{a}^{b} p(t) q(t) \, dt \)

a weighted integral  \( \int_{a}^{b} p(t) q(t) \, dw(t) \)

a discrete weight  \( \int_{a}^{b} p(t) q(t) \, dw(t) = \sum_{i=1}^{n} p(\lambda_i) q(\lambda_i) w_i \)

If two polynomials are orthogonal with respect to an integral  \( \int_{a}^{b} dw(t) \), then we'll often call this the measure that they are orthogonal with respect to.
SEQUENCES AND FAMILIES OF ORTHOGONAL POLYNOMIALS

We are often concerned with a sequence or family of orthogonal polynomials. We will index these by the degree of a polynomial, so that we have the sequence in order of increasing degree. Let \( p_k(t) \) be the polynomial of degree \( k \). By convention, we take \( p_{-1}(t) = 0 \) and \( p_0(t) = c \) for some constant.

Thus, we have the following sequence of orthogonal polynomials that are orthogonal with respect to \( \int_{-1}^{1} dt \):

\[
\begin{align*}
 p_{-1}(t) &= 0 \\
p_0(t) &= 1 \\
p_1(t) &= t \\
p_2(t) &= \frac{1}{2}(3t^2 - 1) \\
p_3(t) &= \frac{1}{2}(5t^3 - t) \\
p_4(t) &= \frac{5}{8}(35t^4 - 30t^2 + 3)
\end{align*}
\]

This sequence is called the Legendre polynomials. Other popular families are:

<table>
<thead>
<tr>
<th>Family</th>
<th>Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Legendre</td>
<td>( \int_{-1}^{1} dt )</td>
</tr>
<tr>
<td>Laguerre</td>
<td>( \int_{0}^{\infty} e^{-t} dt )</td>
</tr>
<tr>
<td>Hermite</td>
<td>( \int_{-\infty}^{\infty} e^{-t^2} dt )</td>
</tr>
<tr>
<td>Chebyshev (1st kind)</td>
<td>( \int_{-1}^{1} \frac{1}{\sqrt{1-t^2}} dt )</td>
</tr>
<tr>
<td>Chebyshev (2nd kind)</td>
<td>( \int_{-1}^{1} \sqrt{1-t^2} dt )</td>
</tr>
<tr>
<td>Jacobi</td>
<td>( \int_{-1}^{1} (1 - t)^{\alpha} (1 + t)^{\beta} dt )</td>
</tr>
<tr>
<td>Gegenbauer</td>
<td>( \int_{-1}^{1} (1 - t^2)^{a-1/2} dt )</td>
</tr>
</tbody>
</table>

Note that these can be scaled and shifted to arbitrary intervals \([a, b]\) too.

ORTHONORMAL POLYNOMIALS

Yes, there are orthonormal polynomials too! Checkout Gautschi's book for more on the relationship. If \( \int_{a}^{b} dw(t) \) is the measure, the idea is that we need \( \int_{a}^{b} p(t)^2 dw(t) = 1 \) in order to get orthonormal polynomials.

2 THE THREE TERM RECURRENCE & TRIDIAGONAL MATRICES

The following fact, which we will not prove, is profound:

**THEOREM 2** Any sequence of orthogonal polynomials of increasing degree satisfies a three-term recurrence and any three-term recurrence defines a sequence of orthogonal polynomials.

**EXAMPLE 3** Consider the Legendre polynomials (described above). They satisfy:

\[
p_{k+1}(t) = \frac{2k + 1}{k + 1} tp_k(t) - \frac{k}{k + 1} p_{k-1}(t).
\]

For instance,

\[
p_2(t) = \frac{2 + 1}{1 + 1} t \left( \frac{t}{p_1(t)} \right) - \frac{1}{1 + 1} \frac{1}{p_1(t)}.
\]

The general form of the three-term recurrence is:

\[
p_{k+1}(t) = \mu_k p_k(t) - \gamma_k t p_k(t) + \eta_k p_{k-1}(t),
\]

where the constants \( \mu_k, \gamma_k, \eta_k \) may depend on \( k \). What this recurrence means is that if we have any sequences of numbers \( \mu_k, \gamma_k, \eta_k \), then they give rise to a set of polynomials that is orthogonal with respect to some measure.\(^3\)

\(^3\) In the notes I'm writing this from, \( \mu_k \) is \( \rho_k \), so beware of future typos.
FROM A THREE-TERM RECURRENCE TO A TRIDIAGONAL MATRIX

It’s this three term recurrence that brings us back to matrix computations, and specifically tridiagonal matrices. Consider the Legendre family. Another way to write the recurrence is:

\[ p_{-1}(t) = 0 \quad p_0(t) = 1 \quad p_1(t) = t \quad (k + 1)p_{k+1}(t) = (2k + 1)tp_k(t) - kp_{k-1}(t). \]

This recurrence forms a matrix:

\[
\begin{bmatrix}
1 & & & \\
-t & 1 & & \\
1 & -3t & 2 & \\
0 & \ddots & \ddots & \ddots \\
\vdots & \ddots & \ddots & \ddots \\
\end{bmatrix}
\begin{bmatrix}
p_0(t) \\
p_1(t) \\
p_2(t) \\
p_k(t) \\
\vdots \\
p_{k+1}(t)
\end{bmatrix}
= 
\begin{bmatrix}
1 \\
0 \\
0 \\
0 \\
\vdots \\
0
\end{bmatrix}
\]

Thus, for instance, we can evaluate a sequence of orthogonal polynomials at a point \( t \) by constructing \( T(t) \) and solving:

\[ T(t)p = e_1. \]

Consequently, any tridiagonal matrix corresponds to a set of orthogonal polynomials.

**There’s a Matlab demo of this in the *orthopoly.*m function!**

There are additional relationships with a matrix called the Jacobi matrix:

\[
J = \begin{bmatrix}
\alpha_1 & 1 & & \\
\beta_1 & \alpha_2 & 1 & \\
\ddots & \ddots & \ddots & 0
\end{bmatrix}
\]

but you’ll need to read more about that in Gautschi’s book.

3 POLYNOMIALS AND MATRICES

Note that if we have a univariate polynomial \( p(t) = \sum_{i=0}^{n} p_i t^i \), then we can evaluate that polynomial with a square matrix argument:

\[ p(A) = \sum_{i=0}^{n} p_i A^i. \]

4 POLYNOMIALS AND ITERATIVE METHODS

First, let us introduce the idea of using polynomials and iterative methods. This has been a homework or exam problem in the past, so it’s worth understanding the details! Consider a Krylov subspace method. The \( k \)th iterate \( x_k \) is in \( \mathbb{R}_k(A, b) = \text{span}\{b, Ab, \ldots, A^{k-1}b\} \). This means that there is some polynomial such that:

\[ x_k = s_{k-1}(A)b. \]

Now, if \( x_k \) is determined by a polynomial \( s_{k-1}(t) \) of degree \( k - 1 \), this means the residual at the \( k \)th step is determined by a polynomial of degree \( k \):

\[ r_k = b - Ax_k = b - As_{k-1}(A)b = \left( I - As_{k-1}(A) \right)b. \quad \text{Thus} \quad r_k = p_k(A)b \]

where the polynomial \( p_k(t) = 1 - ts_{k-1}(t) \) has degree \( k \).

We call these two polynomials:

\[ s_k(t) \quad \text{the solution polynomial} \]
\[ p_k(t) \quad \text{the residual polynomial}. \]

It turns out that the residual polynomial already must have some special structure! Note that \( p_k(t) \) is defined to be equal to \( 1 - ts_{k-1}(t) \). Thus, \( p_k(0) = 1 \). So any residual polynomial must evaluate to 1 at the value \( t = 0 \). As a matrix statement, this means:

\[ p_k(0)b = b. \]

3
5 ORTHOGONAL POLYNOMIALS AND ITERATIVE METHODS

Thus far, we haven’t run into orthogonal polynomials yet. But let’s design an iterative method with a fairly natural property using orthogonal polynomials.

**Design goal**  We want the \( k \)th residual from the iterative method to be orthogonal to all previous residuals. Or more formally, \( r_j^T r_j = 0 \) for \( j < k \). This goal is equivalent to the idea that our residual should always include new information at each step and should never include information we could have factored out.

*Orthogonal polynomials will help us achieve this goal!*  

Let’s state what we have:

\[
r_k = b - Ax_k = p_k(A)b.
\]

We want:

\[
r_j^T r_j = b^T p_k(A)^T p_j(A)b = 0.
\]

Thus, if we *create* an orthogonal polynomial \( p_k(t) \) where \( p_k(0) = 1 \) and

\[
\int p_k(t) p_j(t) \, dw(t) = (r_k)^T r_j,
\]

we will implicitly create an iterative method where the residuals are orthogonal.\(^4\)

*The three term recurrence helps us do this!*  

Recall:

\[
p_{k+1}(t) = \mu_k \left( p_k(t) - y_k t p_k(t) \right) + \eta_k p_{k-1}(t).
\]

So if we can determine \( \mu_k, y_k, \eta_k \) from our constraints, then we’ll be able to figure out what the next residual polynomial is.

Let’s enumerate our constraints:

(i) \( p_{k+1}(0) = 1 \)

(ii) \( r_{k+1} = p_{k+1}(A)b = \mu_k \left( r_k - y_k Ar_k \right) + \eta_k r_{k-1} \)

(iii) \( (r_{k+1})^T r_j = 0, j < k \).

Constraint (i) implies:

\[
1 = p_{k+1}(0) = \mu_k + \eta_k \quad \Rightarrow \quad \eta_k = 1 - \mu_k.
\]

Thus, we now have a revised constraint (ii):

\[
r_{k+1} = p_{k+1}(A)b = \mu_k \left( r_k - y_k Ar_k \right) + (1 - \mu_k) r_{k-1}.
\]

If we apply constraint (iii) with \( j = k \), we have:

\[
0 = (r_k)^T r_{k+1} = \mu_k \left( (r_k)^T r_k - y_k (r_k)^T Ar_k \right) + (1 - \mu_k) (r_k)^T r_{k-1}
\]

or

\[
y_k = \frac{r_k^T r_k}{r_k^T Ar_k}.
\]

Finally, using constraint (iii) with \( j = k - 1 \), we can solve for\(^5\)

\[
\mu_k = \frac{r_{k-1}^T r_{k-1}}{r_{k-1}^T r_{k-1} + y_k r_{k-1}^T Ar_k}.
\]

This lets us compute \( r_{k+1} \)

\(^4\) Take a moment to understand what is going on here, as it’s a key step. We are first noting that \( r_k \) can be expressed as a polynomial in \( A \). We are now saying, let’s control that polynomial to achieve our goal! But we’ll have to obey some constraints to make it work.

\(^5\) Check for an error here... and this simplifies more, see Saad.
GETTING THE SOLUTION POLYNOMIAL

Thus far, we have the residual polynomial \( r_k = p_k(A)b \). We do, however, need to recreate the solution too! To do so, we note that

\[
s_k(t) = \frac{1 - p_k(t)}{t} = \frac{1 - \mu_k(p_k(t) - y_k p_k(t)) - (1 - \mu_k)p_{k-1}(t)}{t}.
\]

Now we add and subtract \( \mu_k/t \) in order to rewrite this as:

\[
s_k(t) = \mu_k \left[ \frac{1 - p_k(t)}{t} - y_k p_k(t) \right] - \frac{(1 - \mu_k)}{t} \frac{1 - p_{k-1}(t)}{t}
\]

\[
= \mu_k(s_{k-1}(t) - y_k p_k(t)) - (1 - \mu_k)s_{k-2}(t).
\]

Hence, we have:

\[
x_{k+1} = \mu_k(x_k - y_k r_k) - (1 - \mu_k)x_{k-1}.
\]

6 A POLYNOMIAL FORM OF CONJUGATE GRADIENT

In a small surprise, we’ve arrived at a new form of the conjugate gradient algorithm! The iterates generated by this method are mathematically equivalent to those generated by CG! The way to prove this is to show that the residuals constructed in CG automatically satisfy the same property and live in the same subspace, hence, they must be the same.