The Transform Method in general

Stitch 12/7/21

This document uses a LaTeX-template in Microsoft Word called WordTEX[1].

1 Introduction – What is the Transform Method?

1.1 Background

We use Transforms to make differential equations simpler to solve, and in the Transform Method that I describe in detail in this document, a linear first order ODE that is easily solved. The Transform method can be used on 2nd order linear partial differential equations. The examples I will use involve a PDE that is 2nd order with respect to x and 1st order with respect to time t. This method works for both constant- and variable-coefficients with roughly equal ease.

1.2 Prerequisites, for readers and for the problems

Readers should first go through the [01 Stitch Notes – OPDE Review.pdf], focusing on 2 – separation of variables, 6 – Interesting/Important Linear Algebra, and 7 – Superposition Principle & Fundamental Solutions for an overview of the typical method used to solve PDEs, and the basics of eigen-functions, -vectors and -values.

The Transform Method itself (as far as my examples go) requires a few things:

1. The problem should be solvable by separation of variables
2. The problem should be of the first degree with respect to time (although it is theoretically possible to use the method on equations with higher degrees wrt time).
3. We need well-defined boundary and initial conditions to find a specific solution. A general solution will be possible without these, as usual.

[1] WordTEX can be installed from here: *https://www.andrew.cmu.edu/user/twildenh/wordtex/*. Without the template and fonts, the document will appear less LaTeX-y but should still be readable. The attached PDF should be formatted properly for read-only access.

1.3 A brief overview of the transform method and deriving suitable transform pairs

In the examples I encounter, the Transform method has always required turning a partial DE into an ordinary DE first by expressing the x-portion as a Sturm-Liouville Problem. S-L problems are easier to solve, but they almost always require knowing eigenfunctions to some linear operator. An example of this expression may be

$$q_t(x,t) - q_{xx}(x,t) = 0 \tag{1.1}$$

such that
$q(x,0) = q_0(x)$
$q(0,t) = Kq_x(0,t)$
$q(1,t) = Kq_x(1,t)$

Suppose we had some magic transform function $T[\cdot](j)$ that is linear and we apply it to both sides of the above so that we can get

$$T[q_t(x,t)](j) - T[q_{xx}(x,t)](j) = 0$$

Suppose also that this magic function can turn second derivatives into non-derivative functions, so that $T[q_{xx}(x,t)](j) = T[-\lambda q(x,t)](j)$. This supposition only requires us to solve… an S-L problem like this!

$$X_{xx}(x) + \lambda X(x) = 0, \tag{1.2}$$

such that
$X(0) = KX_x(0)$
$X(1) = KX_x(1)$

The DE here can easily be represented as $X_{xx}(x) = -\lambda X(x)$, which looks exactly like the eigenvalue/function equation $\boldsymbol{AX} = \lambda \boldsymbol{X}$, where $\boldsymbol{A}, \boldsymbol{X}$ are a matrix and vector respectively and λ is an eigenvalue. In this case, $\boldsymbol{A}$ is the Linear Transformation, and applying this transform to a vector $\boldsymbol{X}$ is equivalent to multiplying the vector by the eigenvalue λ. The trick is to see that all such S-L problems are basically just the same: we can suppose some differential operator $\mathcal{L}[X](x) = X_{xx}(x)$ that applies to the (eigen)function X and returns its second derivative, AND that it can be expressed as some eigenvalue λ multiplied by $X(x)$.

Seeing this, we only need to find some magic function so that $T\{\mathcal{L}[X](x)\}(j) = T\{-\lambda X(x)\}(j)$. One good way to do this is if the magic transform function $T[\cdot](j)$ is a projection/inner product. Since we can only project the original 'input' ($\mathcal{L}[X](x)$) on one function at a time, we would need to project it on infinitely many eigenfunctions to fully represent the original real 'input'. Looking back at the basics of projections, this is easily done by using infinitely many eigenfunctions that are orthogonal to each other.

This also makes clear what j is: changing it means we are using a different eigenfunction to project on. All we need to find now are the correct eigenfunctions to represent it properly. Once we have the eigenfunctions, we can derive the correct transform pair by making sure we get the original function back if we perform $T^{-1}\left\{\left(T[\phi(x)](j)\right)_{j=0}^{\infty}\right\}(x) = \phi(x)$ (inverse transform on the infinite sequence given by the forward transform). The pair refers to mainly the correct inner product for the forward transform and the correct coefficients on each of the infinitely many terms for the inverse transform.

The examples below will use the following general 4 steps to apply the transform method:

1. Express the x part of the PDE as a linear operator $\mathcal{L}$ (e.g. $q_t - q_{xx} = 0 \Rightarrow \mathcal{L}[q] = -q_{xx}$). Then, apply the appropriate inner product to the linear operator, and integrate by parts to derive the Sturm-Liouville Problem that will later give us eigenfunctions E_j (e.g. $\mathcal{L}[E_j] = \lambda E_j, [E_j q_x - E_j' q]_0^1 = 0$)
2. Use these Eigenfunctions and work backwards from our preferred inverse transform definition, to get the correct weights for our weighted infinite sum. (The inverse transform *is* the weighted infinite sum)
3. Solve the whole PDE by transforming it so we end up with a simpler problem wrt t alone.

It is useful to learn how to derive the correct eigenfunctions and transform pair through a few examples. In all the following examples, the general gist is that we solve the S-L problem to find our class(es) of eigenfunctions and then find the correct coefficients for the inverse transform sum.

2 Example – The Robin Heat Equation (self-adjoint)

2.1 Defining the Linear Operator and finding the S-L Problem

Suppose the problem posed earlier (Equation (1.1)): $q_t(x,t) - q_{xx}(x,t) = 0$. We are going to disregard the boundaries and initial conditions for now, including them as we need them.

As mentioned before, we want some magic transform to solve this problem for us. In this problem we use the L^2 inner product $\langle A, B\rangle = \int_0^1 AB \ dx$ and we want to find some eigenfunction E_j (it depends on j because our transform outputs a sequence indexed by j remember?) such that

$$\langle q_t, E_j\rangle + \langle -q_{xx}, E_j\rangle = 0$$

$$\text{and} \quad \langle -q_{xx}, E_j\rangle = \lambda_j \langle q, E_j\rangle$$
$$\text{Which is equivalent to} \quad \langle \mathcal{L}[q], E_j\rangle = \lambda_j \langle q, E_j\rangle$$

Luckily, $\mathcal{L}[\phi](x) = \phi''(x)$ is self-adjoint so $\langle \mathcal{L}[q], E_j\rangle = \langle q, \mathcal{L}[E_j]\rangle$. The other non-self-adjoint case is discussed later.

If we are able to do this, we reduce our 2nd order PDE into a 1st order ODE wrt time t![2] To find a suitable E_j we look at what this inner product would actually be:

$$\int_0^1 q_{xx} E_j \ dx = [E_j q_x - E_j' q]_{x=0}^{x=1} + \int_0^1 E_j'' q \ dx$$

This gives us 2 requirements to fulfil for our transform to work.

1. The square parentheses term needs to disappear. After using our original PDE boundary conditions and then some simplification (see 04 – Robin Heat – Transform.pdf for the full solution), we need

$$E_j(0) = K E_j'(0)$$
$$\text{and } E_j(1) = K E_j'(1)$$

[2] We just need to inverse transform $\langle q_t, E_j\rangle + \lambda_j \langle q, E_j\rangle = 0$ and we get our 1st order ODE in time.

2. The last integrand needs to be expressible in terms of q and E_j only (without any derivatives). This means requiring that

$$E_j''(x) = \lambda_j E_j(x)$$

Putting these 2 requirements together we can get our S-L problem[3], and it is solved using easily found eigenfunctions $E_j(x)$. With this, we have our forward transform applied on function $\phi(x)$:

$$T[\phi(x)](j) = \int_0^1 \phi(x) E_j(x) \ dx$$

2.2 The correct weights for the inverse transform

Now, we need to think about the inverse transform. Since we found E_j through the S-L problem, they should be orthogonal to each other, if the linear operator $\mathcal{L}$ is self-adjoint[4], which it is for the Robin Heat equation. Now, we know that the inverse transform function $T^{-1}[\cdot](x)$ should be a weighted sum such as

$$T^{-1}[T[\phi(\cdot,t)](j)] \equiv \phi(\cdot,t) = \sum_{j=0}^{\infty} D_j E_j(x)$$

Following the steps as laid out in [03 Dave's Transform.pdf], we can apply the forward transform to the second and third part of the equation, and use the linearity of the transform to get

$$T[\phi(\cdot,t)](k) = \sum_{j=0}^{\infty} D_j T[E_j(x)](k) \tag{2.1}$$

Where $T[E_j(x)](j) = \int_0^1 E_k(x) E_j(x) \ dx$.

We know (mostly because we have already "done" the separation of variables approach) that $E_j \perp E_k$ iff $j \neq k$. This can be verified by simply performing the inner product (since we already know E_j explicitly). By performing the inner product we also realise that the inner product is non-zero when $j = k = 0$ and when $j = k \neq 0$, and 0 otherwise. See the top of page 4 of [04 Robin

[3] We get exactly the same S-L problem (1.2) as shown on page 2.

[4] Self-adjoint, for now, just means that both the forward and inverse transforms use the same kernel/eigenfunction $E_j(x)$. I go into this topic in the later examples where we encounter non-self-adjoint linear operators.

Heat – Transform.pdf] for the derivation of this result. Remembering that K was defined by our boundary conditions, we get that

$$\Rightarrow T[E_j(x)](k) = \begin{cases} 0, & j \neq k \text{ and eigenfunctions orthogonal} \\ \dfrac{Ke^{2/K} - K}{2}, & j = k = 0 \\ \dfrac{1}{2}[(Kj\pi)^2 + 1], & j = k \neq 0 \end{cases}$$

Substituting this into Equation (2.1) and making D_j the subject gives us

$$D_j = \begin{cases} \dfrac{2T[\phi](k)}{Ke^{2/K} - K}, & j = k = 0 \\ \dfrac{2T[\phi](k)}{(Kj\pi)^2 + 1}, & j = k > 0 \end{cases}$$

Thus, we now have both our forward transform (aka inner product) and the inverse transform (the infinite sum whose weights are now known).

2.3 Solving the PDE by transforming it

After applying the transform to both sides of the equation, we get

$$\frac{d}{dt}T[q(\cdot,t)](j) - \lambda_j T[q(\cdot,t)](j) = 0$$

Seeing that this is just a 1st order ODE in time, the solution, using the initial condition $q(x,0) = q_0$ is

$$T[q(\cdot,t)](j) = T[q_0](j)e^{\lambda_j t}$$

(Try substituting $f(t) = T[q(\cdot,t)](j)$ if this is not clear).

Now we notice that D_j has $T[\phi](j)$ in its numerator, and we have just found it to be equal to $T[q_0](j)e^{\lambda_j t}$. So, our solution requires simply reconstructing $q(x,t)$ using an infinite sum:

$$q(x,t) = \sum_{j=0}^{\infty} D_j E_j(x)$$

Where

$$D_j = \begin{cases} \dfrac{2T[q_0](j)e^{\lambda_j t}}{Ke^{2/K} - K}\ , & j = 0 \\ \dfrac{2T[q_0](j)e^{\lambda_j t}}{(Kj\pi)^2 + 1}, & j > 0 \end{cases}$$

And $E_j(x)$ being the solution to the S-L problem we solved earlier.

3 Example – Advection Diffusion Problem – Non-self-adjoint

In this section we consider the following problem for arbitrary fixed $c \in \mathbb{R}$ and $(x,t) \in (0,1) \times (0,T)$

$$q_t(x,t) - q_{xx}(x,t) + cq_x(x,t) = 0 \tag{1.1}$$

subject to

$$q(x,0) = q_0(x) \text{ for } x \in [0,1]$$

I will disregard the exact boundary conditions so the method is generally applicable. As before, we will follow the 3-step process. The key learning point here is that non-self-adjoint problems will require 2 'versions' of eigenfunctions E_j^* and E_k $(j \neq k)$ such that $E_j^* \perp E_k$ but $E_j^* \neg\perp E_k^*$ and likewise for E_j. Thus, the forward transform uses E_j^* as an eigenfunction and the inverse transform uses E_j. If you have understood the previous example, it may be clear already that we have to do this because we want to find the weights D_j and this requires the infinite sum to collapse into 1 term (when $j = k$) which in turn requires that $E_j \perp E_k^*$.

3.1 Finding the Linear Operator and setting up the S-L Problem

Here, our linear operator is $\mathcal{L}[q] = -q_{xx} + cq_x$. Taking inner product with some eigenfunction $E_j^*(x)$, we realise after some integration by parts that

$$\langle E_j^*(x), \mathcal{L}[\phi(x)] \rangle = [-E_j^*(x)\phi'(x)]_0^1 + \langle -E_j^{\prime\prime *}(x) - cE_j^{\prime *}(x), \phi(x) \rangle$$

2 things/requirements pop out of this relation:

1. The operator $\mathcal{L}$ is non-self-adjoint, as we have to define $\mathcal{L}^* E_j^* = -E_j^{\prime\prime *}(x) - cE_j^{\prime *}(x)$. In simpler words, the yellow highlighted terms above are not outputs of the same linear operator.
2. The S-L problem would require $[-E_j^*(x)\phi'(x)]_0^1 = 0$ and $-E_j^{\prime\prime *} - cE_j^{\prime *} = \lambda_j E_j^*$.

The S-L problem can be solved to find E_j^*.

3.2 Inverse Transform Derivation

Now that we know how to perform the forward transform using $T[\phi(x)](j) = \int_0^1 \phi E_j^* \ dx$, we move to finding the inverse transform which is by definition

$$\phi(x) = \sum_{j=0}^{\infty} D_j E_j(x)$$

We can find the weights in exactly the same way as before (transform both sides with index k, collapse the infinite sum on the RHS into a single term IFF $j = k$). The key to understanding the whole 'adjoint' problem will be

$$T[E_j(x)](k) = \int_0^1 E_k^*(x) E_j(x) \ dx$$

Which is non-zero IFF $k = j$ and 0 otherwise. If we had used the same eigenfunction (as if we had a self-adjoint problem) for both the forward and inverse transform, they would not be orthogonal to each other and the infinite sum would not collapse! With this, we can firstly find the weights D_j and then transform the whole problem now to get the solution that we have on page 5 of [06 AdvDiff – Transform.pdf].

4 Example – Bessel RSP – Variable coefficient PDE, self-adjoint

For my final example, I go through a variable coefficient problem, which is self-adjoint. The key takeaway from this example is that the same method as above can be applied wholesale with no additional difficulty.

The problem to solve is that for arbitrary fixed $m \in \mathbb{N}_0$ and $(x,t) \in (0,1) \times (0,T)$

$$q_t(x,t) - \left(q_{xx} + \frac{1}{x}q_x - \frac{m^2}{x^2}q\right)(x,t) = 0 \tag{1.1}$$

subject to

$$q(x,0) = q_0(x) \text{ for } x \in [0,1]$$

and some suitable boundary conditions.

My full solution to this problem has a few learning prerequisites:

1. The problem has its 'x-component' in the form of Bessel's Differential Equation. Bessel's DE has Bessel functions of the first kind as its eigenfunctions.
2. Bessel functions of the first kind form a basis of functional space over which any function can be expressed (similar to trigonometric bases of functional spaces).
3. This new functional space has a new type of inner product $\langle f,g\rangle_A = \int_0^b xf(x)g(x)\ dx$, using which, Bessel functions of the first kind are orthogonal to each other given different indices.

4.1 Deriving the S-L problem

Our linear operator here would be $\mathcal{L}[\phi](x) \equiv \left[\phi'' + \frac{1}{x}\phi' - \frac{m^2}{x^2}\phi\right](x) = \frac{1}{x}\left[\frac{d}{dx}\left(x\phi'(x)\right) - \frac{m^2}{x}\phi(x)\right]$. The latter form is used because it clearly shows the need for a different kind of inner product to make eigenfunctions orthogonal.

With the A-inner product, we also get the self-adjoint relation: $\langle \mathcal{L}\phi, E_j\rangle_A = \langle \phi, \mathcal{L}E_j\rangle_A$, IFF $\left[xE_j(x)\phi'(x) - xE_j'(x)\phi(x)\right]_0^1 = 0$. This, together with 1 more boundary condition and that $\mathcal{L}E_j = \lambda_j E_j$ gives us our S-L problem that we know is solved by Bessel functions of the first kind.

Thus, we have our forward transform without too much difficulty.

4.2 The inverse transform weights

Again, we try to find weights c_j such that

$$\phi(x) = \sum_{j=0}^{\infty} c_j J_m(\lambda_j x)$$

Given that the Bessel functions of the first kind are orthogonal to each other IF their arguments have different λ_j and λ_k, we have again that

$$c_j = \frac{\langle \phi, J_m(\lambda_j x) \rangle_A}{\|J_m(\lambda_j x)\|_A^2} = \frac{T_B[\phi](j)}{\|J_m(\lambda_j x)\|_A^2}$$

4.3 Solving the whole equation

Yet again, we have no trouble applying the transform to the original problem to get

$$\frac{d}{dt} T[q(\cdot, t)](j) + \lambda_j T[q(\cdot, t)](j) = 0$$

Which is a first order ODE in time, and then we get our complete solution, just as before, as presented in [08 Bessel RSP – Transform.pdf].

5 Conclusion

The transform method has a few clear and important benefits:

1. We can solve PDEs as long as we know the eigenfunctions to the 'more complicated' portion of the PDE. For example, we knew the eigenfunctions for $\mathcal{L}q = -q_{xx}$ which solved the Heat Equation easily. The $-q_{xx}$ is what I call the 'more complicated portion' in the spatial parameter since we have otherwise a simple 1st order ODE in time. We can focus on the tougher portion first, and once we have solved it, the rest is trivial.
2. The self-adjoincy issue is present in the Separation of Variables method too, and in that sense this method has no clear edge, except that here we can work using abstract E_j first and realise that the problem is non-self-adjoint. In the separation of variables method, any eigenfunctions found (in closed form) have to be verified to be orthogonal, otherwise we may miss it out and wrongly assume to have found a solution. The Transform Method thus is less error-prone.
3. We do not have to assume at the beginning in the Transform Method that the solution is of the form $q(x,t) = X(x)G(t)$, like we have to do in Separation of Variables.

It also has a few drawbacks, but rarely those that separation of variables does not also have:

1. We still have to solve an S-L problem in the middle of the Transform Method, which is analogous to that in the Sep. of Vars. Method. Tedium has not been reduced in that portion.
2. We have to find the correct inner product to use which will let the eigenfunctions be orthogonal to each other. This may well be a simple matter if we try the L^2-inner product and it fits, or if we know beforehand that the Bessel functions of the first kind are A-inner product space orthogonal. But if we have no knowledge about the eigenfunctions we have found, it may be difficult to land upon the correct inner-product to use. One theory that professor Dave had is that we can find an inner product that allows us some kind of self-adjoint relation e.g. $\langle \mathcal{L}\phi, E_j \rangle_A = \langle \phi, \mathcal{L}E_j \rangle_A$. That is again, a requirement-based guess and verify process (I may be wrong here) which is still not optimal in my opinion.