NTD Thermal Model

May 25, 2017

1 NTD Event Thermal Modeling Calculation

1.1 Intro

For bolometer, K is the thermal conductivity of various resistances, such as ones found in glue or wires. It is related to temperature by a power law relation: $G(T) = G_0 T^{\beta}$. It can also be written as the time derivative of power $\frac{dP}{dt}$, such that we can express power as $P(T) = \int_{T_S}^{T} G(T')dT'$, where T_s is the temperature of the heat sink. Using this, we can integrate the equation to get

$$(T^{\beta+1} - T_s^{\beta+1}) = \frac{\beta+1}{G_0} P(T)$$

In the small limit theorem ($\Delta T \ll T$) and if T_s is constant, then we get

$$C\frac{dT}{dt} + K\Delta T = \Delta P$$

1.2 Vivek's Thermal Model

1.2.1 Power Conservation Equations: NTD Event

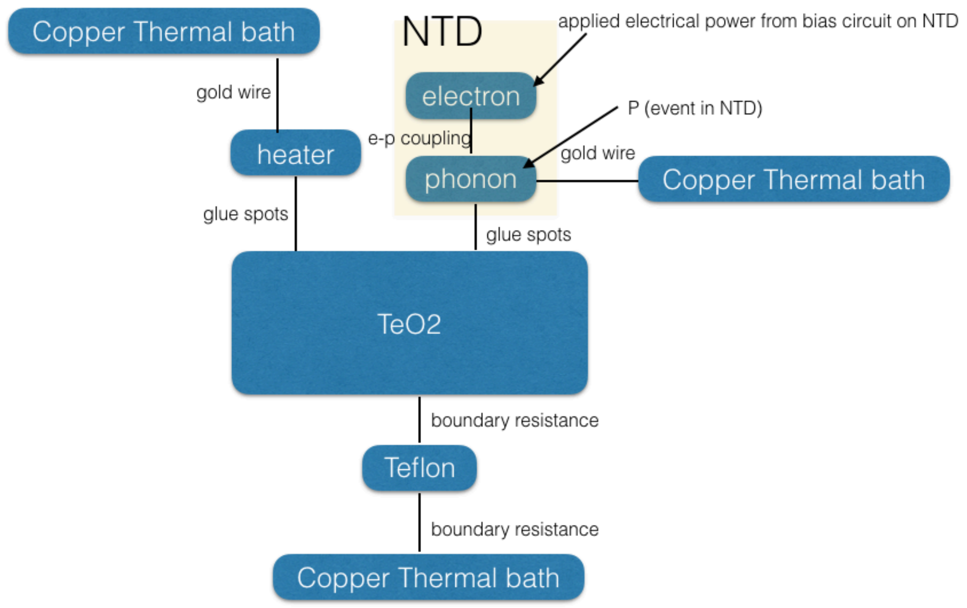
Heater: $C_{Heater}\frac{dT_{Heater}}{dt} + K_{HeaterGoldWire}(T_{Heater} - T_{Bath}) - K_{HeaterGlue}(T_{TeO_2} - T_{Heater}) = 0$

Teflon: $C_{Teflon}\frac{dT_{Teflon}}{dt} + K_{Teflon\leftrightarrow bath}(T_{Teflon} - T_{Bath}) - K_{TeO_2\leftrightarrow Teflon}(T_{TeO_2} - T_{Teflon}) = 0$

Crystal: $C_{Crystal}\frac{dT_{Crystal}}{dt} + K_{HeaterGlue}(T_{TeO_2} - T_{Heater}) + K_{NTDGlue}(T_{TeO_2} - T_{NTDPhonon}) + K_{TeO_2\leftrightarrow Teflon}(T_{TeO_2} - T_{Teflon}) = 0$"

Phonon: $C_{Phonon}\frac{dT_{Phonon}}{dt} + K_{NTDGoldWire}(T_{Phonon} - T_{Bath}) - K_{NTDGlue}(T_{TeO_2} - T_{Phonon}) - K_{E-PCoupling}(T_{Electron} - T_{Phonon}) = P_{NTD_E vent}$

Electron: $C_{Electron}\frac{dT_{Electron}}{dt} - P_{ElectricalPower} + K_{E-PCoupling}(T_{Electron} - T_{Phonon}) = 0$



title

1.3 Solving equations with Fourth Order Runge-Kutta Method

4th Order Runge-Kutta Generator: https://www.codeproject.com/Tips/792927/Fourth-Order-Runge-Kutta-Method-in-Python

```
In [23]: # fourth order Runge-Kutta method in 5 dimensions
         def rK5(a, b, c, d, e, fa, fb, fc, fd, fe, hs):
                 a1 = fa(a, b, c, d, e)*hs
                 b1 = fb(a, b, c, d, e)*hs
                 c1 = fc(a, b, c, d, e)*hs
                 d1 = fd(a, b, c, d, e)*hs
                 e1 = fe(a, b, c, d, e)*hs
                 ak = a + a1*0.5
                 bk = b + b1*0.5
                 ck = c + c1*0.5
                 dk = d + d1*0.5
                 ek = e + e1*0.5
                 a2 = fa(ak, bk, ck, dk, ek)*hs
                 b2 = fb(ak, bk, ck, dk, ek)*hs
                 c2 = fc(ak, bk, ck, dk, ek)*hs
                 d2 = fd(ak, bk, ck, dk, ek)*hs
                 e2 = fe(ak, bk, ck, dk, ek)*hs
                 ak = a + a2*0.5
                 bk = b + b2*0.5
                 ck = c + c2*0.5
```

```
            dk = d + d2*0.5
            ek = e + e2*0.5
            a3 = fa(ak, bk, ck, dk, ek)*hs
            b3 = fb(ak, bk, ck, dk, ek)*hs
            c3 = fc(ak, bk, ck, dk, ek)*hs
            d3 = fd(ak, bk, ck, dk, ek)*hs
            e3 = fe(ak, bk, ck, dk, ek)*hs
            ak = a + a3
            bk = b + b3
            ck = c + c3
            dk = d + d3
            ek = e + e3
            a4 = fa(ak, bk, ck, dk, ek)*hs
            b4 = fb(ak, bk, ck, dk, ek)*hs
            c4 = fc(ak, bk, ck, dk, ek)*hs
            d4 = fd(ak, bk, ck, dk, ek)*hs
            e4 = fe(ak, bk, ck, dk, ek)*hs
            a = a + (a1 + 2*(a2 + a3) + a4)/6
            b = b + (b1 + 2*(b2 + b3) + b4)/6
            c = c + (c1 + 2*(c2 + c3) + c4)/6
            d = d + (d1 + 2*(d2 + d3) + d4)/6
            e = e + (e1 + 2*(e2 + e3) + e4)/6
            return a, b, c, d, e
```

```
In [24]: #define constants
         cap = [1,2,3,4,5]
         k = [1,2,3,4,5,6,7]
         ts = .001
```

```
In [36]: #define equations

         def phonon(a,b,c,d,e):
             return (k[0]*(d-a)+k[1]*(b-a)-k[2]*(a-ts))/cap[0]

         def electron(a,b,c,d,e):
             return (k[1]*(b-a))/cap[1]

         def heater(a,b,c,d,e):
             return (k[3]*(d-c)-k[4]*(c-ts))/cap[2]

         def crystal(a,b,c,d,e):
             return (-k[0]*(d-a)-k[3]*(d-c)-k[5]*(d-e))/cap[3]

         def teflon(a,b,c,d,e):
             return (k[5]*(d-e)-k[6]*(e-ts))/cap[4]
```

```
In [39]: #run algorithm
```

```
        def getValues():
            a,b,c,d,e,hs = 1.0,0.0,0.0,0.0,0.0,0.05
            for i in range(20000):
                a, b, c, d, e = rK5(a, b, c, d, e, phonon, electron, heater, crystal, teflon, h
            print a,b,c,d,e
```

```
In [41]: getValues()
```

```
-2.21615939412e+301 -7.3116477183e+301 -7.86614780129e+299 -2.18102721855e+300 -7.93846507477e+2
```

```
In [ ]:
```