Contents

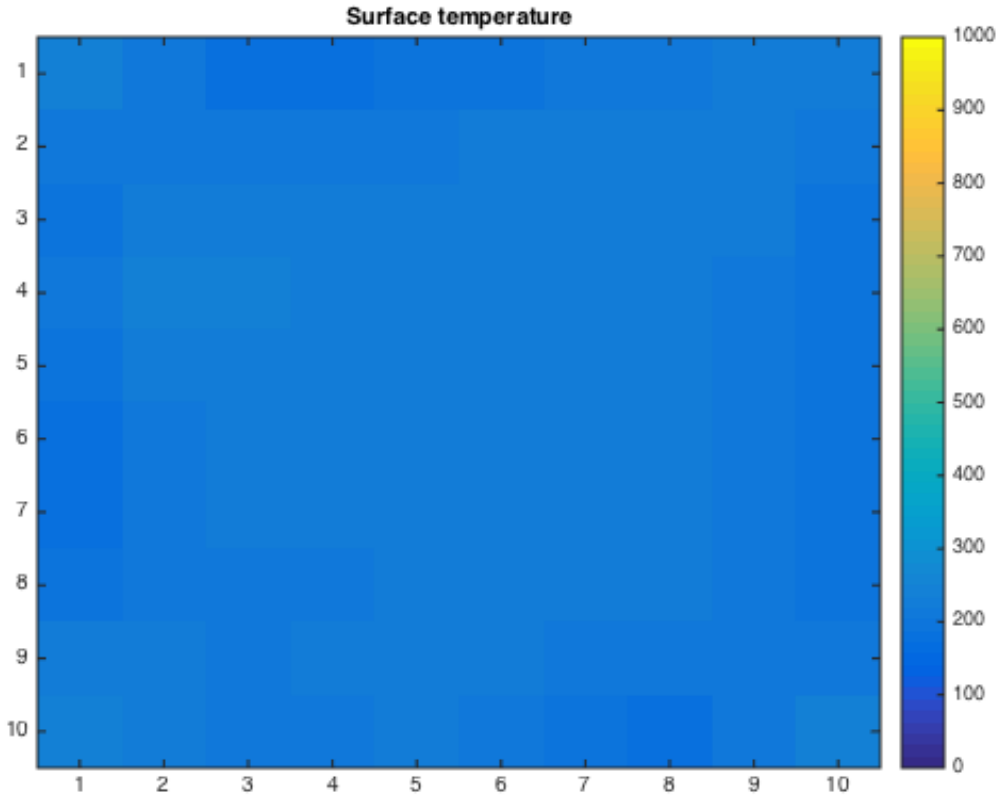
Least Squares

When inputing basis vectors into the simulation function, we saw that different regions of the surface were being heated up. It seemed that there was very little overlap between the functions of the basis vectors. We therefore take the functions of the basis vectors as the columns of an A matrix because they are more or less linearly independant. This turns the problem into a least squares problem with the equation y = Ax. To minimize the cost function we minimize the inside of the square root.

```
clear all;
Tdes = 500;
Tdes_vector = Tdes*ones(100,1); %Desired value at each tile
for i = 1:10
    p = zeros(10,1);
    p(i) = 1;
    A(:,i) = surface_heating_sim(p);  %Simulation of basis vectors
end
xhat = (A'*A)\A'*Tdes_vector %Least squares to find xhat which minimizes cost function
e = sqrt(1/100*norm(surface_heating_sim(xhat)-Tdes_vector)^2);
```

```
Simulating the system..............
Done!
Simulating the system..............
Done!
Simulating the system..............
Done!
Simulating the system..............
Done!
Simulating the system..............
Done!
Simulating the system..............
Done!
Simulating the system..............
Done!
Simulating the system..............
Done!
Simulating the system..............
Done!
Simulating the system..............
Done!

xhat =

    0.4194
    0.4324
    1.3558
    0.9766
    2.7602
    1.1718
```

```
    0.8185
    0.3758
    2.9371
    1.1239

Simulating the system..............
Done!
```



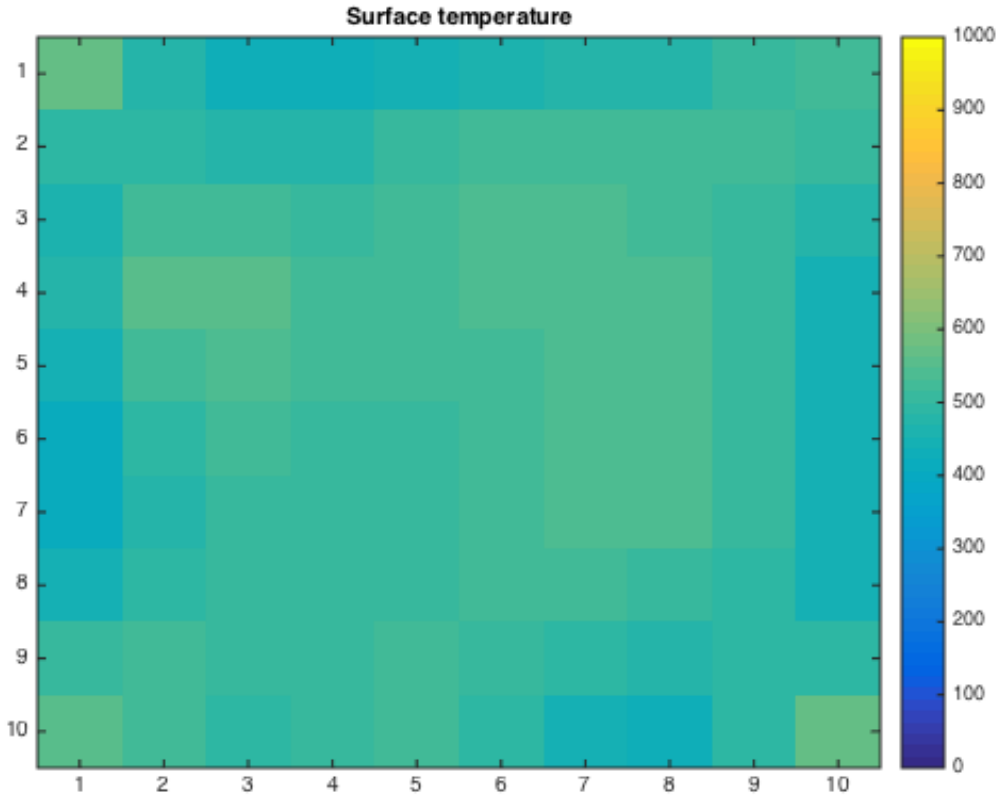
Linearity

We noticed when playing around with the values that the simulation function had certain linear properties. Scaling the input basis vectors by some alpha resulted in linear scaling of the output as well. We saw a similar result when scaling the xhat generated above. We saw that xhat did not correctly minimize the system, it simply gave the ratio of the power lamps to create uniform heating on the surface. To find the appropriate xhat which acheives the goal of heating the surface to a uniform temperatue Tdes we much scale xhat. As stated in the problem the simulation function is nearly affine, therefore we assume a form y = m*c+b. Where m is the slope of xhat, c is the scaling factor, and b is the y intercept. Knowing a Tdes (y in this equation), xhat, and b we can solve for the appropriate m which minimizes the cost function.

```
xhat = xhat/norm(xhat); %normalize xhat
T0 = surface_heating_sim(zeros(10,1));
Txhat = surface_heating_sim(xhat);

% Find the slope, y intercept of the line
m = mean(Txhat) - mean(T0);
b = mean(T0);

% Solve for c
c = (Tdes - b)/ (m);
```

```
%Error using this method
e = sqrt(1/100*norm(surface_heating_sim(c*xhat)-Tdes_vector)^2);

['The RMS error due to using this method is ' num2str(e)]

['The lamp powers used are as follows: ']
c*xhat

% As can be seen above, the surface_heating_sim function was used 12 times
% 10 times to create the A matrix to find xhat and then twice more to
% calculate to generate a model for the line. This is not counting the
% times that the function was used to compute the RMS error.
```

```
Simulating the system..............
Done!
Simulating the system..............
Done!
Simulating the system..............
Done!

ans =

The RMS error due to using this method is 34.7671


ans =

The lamp powers used are as follows:


ans =

    1.0483
    1.0806
    3.3884
    2.4407
    6.8984
    2.9286
    2.0457
    0.9391
    7.3407
    2.8088
```



Published with MATLAB® R2014b