Journal of Applied Engineering Mathematics

Volume 10, December 2023

IMPLEMENTATION OF CURVE-FITTING ALGORITHMS FOR EXTRAPOLATION OF THE INTERFACE TEMPERATURE IN ROTARY FRICTION WELDING

Bryce Harward & Ben Perry

Mechanical Engineering Department
Brigham Young University
Provo, Utah 84602
harwardbrycel@gmail.com

ABSTRACT

Friction welding is capable of quickly joining separate parts together, often at lower temperatures and with less manual work. Some of the issues surrounding friction welding include the quality of the weld due to the temperatures involved and therefore, the material properties. This paper will attempt to use mathematical methods to analyze the temperature at a friction welding joint between two round metal tubes by analyzing temperature values at thermocouples set on the specimen.

NOMENCLATURE

x: $position\ or\ distance\ from\ initial\ weld\ interface$
T: $temperature\ at\ the\ surface\ of\ the\ workpiece$ $(as\ a\ function\ of\ x)$

INTRODUCTION

Rotary Friction Welding (RFW) is a solid-state welding process that may be employed to join a wide variety of materials. Advantages of RFW processes include narrower heat-affected zones, leading to material properties that are generally more consistent with the base material than those yielded by other welding processes. Workpieces used in RFW processes are typically cylindrical and composed of metals, though other geometries and materials are also viable. An important component of RFW is the temperature at the interface of the weld, as it affects material properties such as grain size and yield strength in the completed weld. Determining the temperature at this interface presents several challenges due to the complex nature of RFW. The interface between the two workpieces moves as they are pressed together, producing a transient boundary condition. In addition, the rate of heat input is difficult to define, as it is dependent on many parameters of the weld process. The combination of these two challenges inhibits implementation of the standard heat equation to determine the temperature at the weld interface. A common approach in determining the temperature at the weld interface includes mounting thermocouples to specific locations on the outer surface of one of the workpieces. These thermocouples report temperature data, which can be used to approximate the interface temperature using mathematical models. Predictive models typically involve diligent validation and implement parameters to fit the model to experimental data as accurately as possible. This paper presents the implementation of simple curve-fitting algorithms in predicting the temperature at the weld interface. Data used in this paper was collected from welds produced with thin-walled cylindrical workpieces.

METHODS

Logarithmic Curve Fitting

Logarithmic fitting involves calculating two parameters for a given, discrete value of time using the values reported by the thermocouples. This is done as follows:

$$\alpha = \frac{n\sum(x \cdot T) - \sum x \cdot \sum T}{n\sum x^2 - (\sum x)^2}$$

$$\beta = \frac{\sum x - \alpha \sum x}{n}$$

These values are then implemented in the trendline equation:

$$y = \alpha x + \beta$$

The position of the interface is equal to half the distance by which the two workpieces have been pressed together. The interface temperature can then be extrapolated by setting x equal to the interface position. This process is repeated for each

discrete value of time to produce a plot of the interface temperature as a function of time. As the two workpieces are pressed together, flash is produced and the two thermocouples nearest to the weld interface become embedded in the flash. As such, the temperature values reported by these thermocouples are omitted from the fitting function as the weld interface passes their initial position.

As can be seen in Figure 1 of the conclusion section, the omittance of temperature values reported by thermocouples outside the weld produces strong variation in the logarithmically-extrapolated interface temperature.

Polynomial Curve Fitting

Polynomial-based curve fitting involves the calculation of the coefficients required to produce a polynomial curve that fits a given dataset. These coefficients are calculated as follows:

$$p(x) = p_1 x^n + p_2 x^{n-1} + \ \ldots + p_n x + p_{n+1}$$

The unit standard deviation is calculated to center x at zero and scale the fitting curve:

$$\hat{x} = \frac{x - \bar{x}}{\sigma_x}$$

The fitting curve may be first order through the *n*th order, where n is the size of the dataset. The same approach is implemented of omitting data reported by thermocouples outside of the weld. As seen in Figure 1 of the conclusion section, this fitting algorithm results in a much more conservative estimate of the interface temperature.

CONCLUSIONS

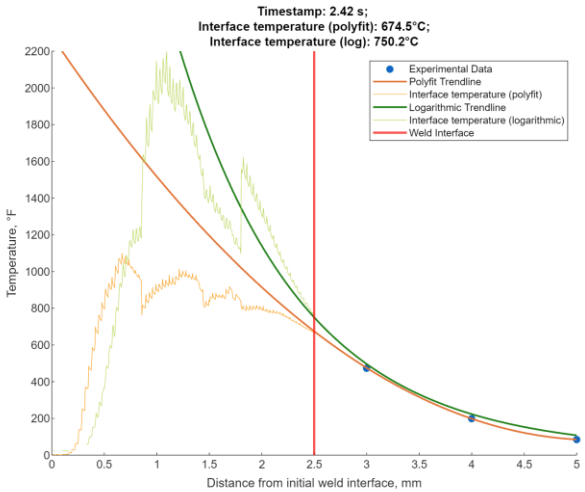

Figure 1. The extrapolated temperature values as a function of time and position, reported by both logarithmic and polynomial curve-fitting algorithms.

The proposed curve-fitting algorithms produce a seemingly appropriate approximation of the interface temperature at face value. Validation of these results could be better analyzed using different approximation methods. Finite-element analysis is a common approach that could be used.

APPENDIX

Matlab script:

```
close all
clear all
clc
tic;
warning('off',
'MATLAB:table:ModifiedAndSavedVarnames');
% Import data from .csv file
data = readtable('10-16-23 600 S011 and S005',
'ReadVariableNames', true);
plotLineWidth = 1.5;
start = 1;
timestamp = 500;
interface = 0;
% Create plots of data
for i = start:length(data.T1)

   % figure(i)
   tempGrad = [data.T1(i) data.T2(i)
data.T3(i) data.T4(i) data.T5(i)];
   tempGrad = tempGrad - data.T1(1);
   position = [1 2 3 4 5];
   if interface > 0.85
       position(1) = [];
       tempGrad(1) = [];
       if interface > 1.8
           position(1) = [];
           tempGrad(1) = [];
       end
   end

   scatter(position, tempGrad,
'MarkerFaceColor', "#0066CC")
   % set(gcf, 'units', 'normalized',
'outerposition', [0 0 1 1])
   xlabel("Distance from initial weld
interface, mm")
   xlim([0 5])
   ylim([0 2200])
   ylabel("Temperature, " + char(176) + "F")
   hold on
   if data.ZPosition_mm_ > -5
       interface =
abs(data.ZPosition_mm_(i))/2;
   else
       interface = -5;
   end
```



```
   yValues = polyfit(position, tempGrad,
length(tempGrad) - 1);

   trendXValues_poly = linspace(0, 5, 100);
   polyVals = polyfit(position, tempGrad,
length(tempGrad) - 1);
   y2 = polyval(polyVals, trendXValues_poly);
   plot(trendXValues_poly, y2, "Color",
"#E8702A", 'LineWidth', plotLineWidth)
   T_int_poly(i - start + 1) =
polyval(polyVals, interface);
   x_int(i - start  + 1) = interface;
   plot(x_int, T_int_poly, "color", "#FFA000")
   logTempGrad = log(tempGrad);
   alpha =
(length(logTempGrad)*sum(logTempGrad.*position
) -
sum(logTempGrad)*sum(position))/(length(logTem
pGrad)*sum(position.^2) - (sum(position))^2);
   beta = (sum(logTempGrad) -
alpha*sum(position))/length(logTempGrad);

   trendXValues_log = linspace(0, 5, 100);
   normalizedYValues = alpha.*trendXValues_log
+ beta;
   trendYValues = exp(normalizedYValues) +
data.T1(1);
   plot(trendXValues_log, trendYValues,
"Color", "#008000", 'LineWidth',
plotLineWidth)
   T_int_log(i - start + 1) =
exp(alpha*interface + beta) + data.T1(1);
   plot(x_int, T_int_log, "color", "#BADA55")
   xline(interface, "Color", "red",
'LineWidth', plotLineWidth + 0.5)
   title("Timestamp: " +
round(data.AdjustedTime_s_(i), 2) + " s; " +
newline + "Interface temperature (polyfit): "
+ round(T_int_poly(i - start + 1), 1) +
char(176) + "C;" + newline + "Interface
temperature (log): " + round(T_int_log(i -
start + 1), 1) + char(176) + "C")
   hold off
   legend("Experimental Data", "Polyfit
Trendline", "Interface temperature (polyfit)",
"Logarithmic Trendline", "Interface
temperature (logarithmic)", "Weld Interface")
   drawnow

exportgraphics(gcf,'testAnimated.gif','Append'
,true);
end
```

