Spring 2019

Interlaminar Damage Detection in Composite Materials

Hariharan Rangarajan
The University of Akron, hr29@zips.uakron.edu

2. Introduction

New commercial jet engines are implementing ceramic-matrix composite (CMC) components as CMCs offer much higher stress-temperature durability than nickel-based super alloys. CMCs have much lower density and can tolerate higher turbine inlet temperatures and reduced cooling needs. Also, these lighter-materials improve the thrust-to-weight capability of jet engines.[1]

The nature of failure and strength degradation in CMCs are different from that of super alloys. It is very important to understand damage progression in these materials and develop techniques to monitor them as they directly relate to operating life of these components in real stress-time-temperature environment.

The Direct Current Potential Drop (DCPD) is an effective non-destructive technique to measure damage in composite materials.[2] DCPD method works by measuring nodal potential differences when a constant DC current flows through the material from one point to another. It requires a minimum of four probes attached to a specimen as shown in Figure 1, to obtain one potential drop data point. Current is flown through the outer probes and potential drop is measured across the inner two leads.

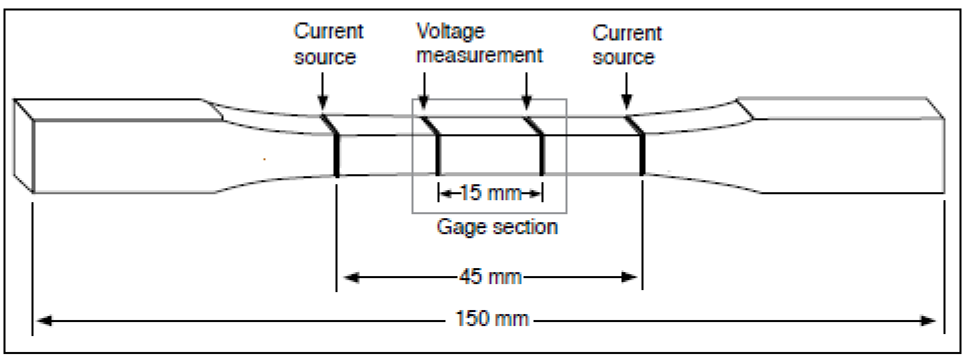


Figure 1: Four-point probe in-plane DCPD measurement setup[1]

Fundamentally, the DCPD technique works based on two principles. One, current flows through the path of least resistance between two points. Two, current spreads slightly beyond the path of least resistance. When implemented correctly, DCPD provides useful data about electrical properties of a material such as in-plane and out-of-plane resistivities. This is important to characterize composite materials such as SiC/SiC (Silicon Carbide) composites since they are anisotropic conductors of electricity (i.e. they have very different in-plane and out-of-plane electrical resistivity). More importantly, potential drop data from DCPD measurements can indicate delamination, voids, surface and near-surface cracks in a material. This report explores the possibility of implementing DCPD in the form of a versatile test equipment to monitor crack growth in composite materials.

3. Experimental Measurement

3.1 Technique

To analyze the feasibility of the potential drop method to detect cracks in composite materials, notched specimen were tensile tested to propagate a crack. The notch created a stress concentration that facilitated crack propagation into the bulk of the specimen during tensile testing. Next the specimen's surface was cleaned to apply nichrome leads. Usually 12-14 lead pairs were applied to obtain adequate potential drop data points. A two-part silver epoxy was mixed in appropriate proportion and applied to the surface as shown in Figure 2, to attach the leads to either sides of the specimen. The epoxy was cured at 100°C for 4hrs. The topmost lead pairs were used to flow a constant DC current from an external power supply. A current of no more than 500mA was applied to avoid heating of the specimen during the experiment. For SiC/SiC specimen (used in this experiment) 100mA provided optimum data quality. The nodal potential drops between subsequent pairs of electrical leads were measured using a multi-channel analog signal recorder. The trend in potential drop was then analyzed to calculate crack length which was then compared to crack length measurements from other damage characterization techniques such as Acoustic Emissions (AE), DIC (Digital Image Correlation) and Microscopy to check consistency in results.

Figure 2: Electrical leads attached to tensile tested specimen[2]

3.2 Data Analysis

The measured raw potential drop data was analyzed and plotted against lead pair position. The trend in potential drop was influenced by the presence of delamination or cracks in the specimen. It Figure 3 it can be observed that potential drop varies linearly with position in a crack region and decreases exponentially through the pristine region. Since this method measured the difference in voltage drop instead of the absolute voltage, the trend observed was independent of the magnitude of current and robust to absolute errors in analog signals.

The data can also be plotted in a semi-log scale to reinforce the exponential trend in potential drop over a pristine region. For instance, in Figure 4 a linear behavior observed in the semi-log plot of potential decay in Carbon Fiber Reinforced Polymer (CFRP) composites indicated exponential decrease in potential drop on a regular plot.

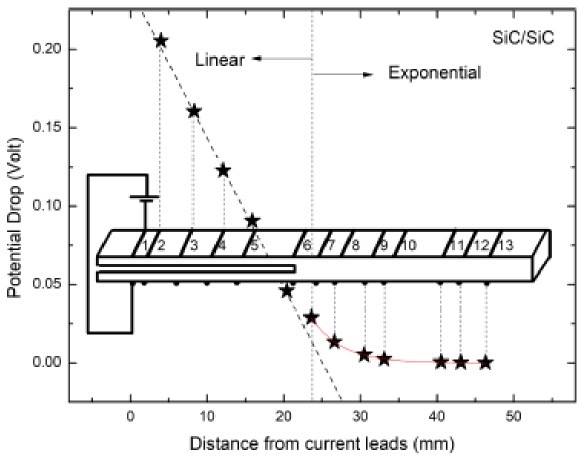


Figure 3: Potential drop as a function of electrical lead position[2]

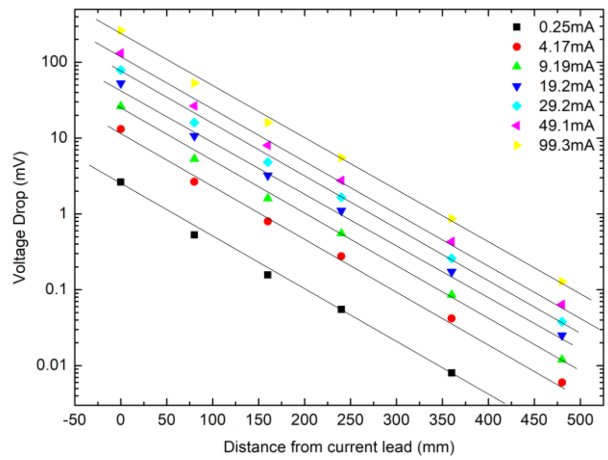


Figure 4: Semi-log plot of potential drop in intact CFRP specimen[2]

4. Modeling Potential Drop in CMCs

Current flow through composite materials can be modeled as a ladder network of resistors as shown in Figure 5. The series resistors represent in-plane resistance and parallel resistors represent out-of-plane resistance of the material. A crack/delamination in the material would be represented as a resistor of very high resistance or a bare node (infinite resistance). Knowing say the out-of-plane resistivity of a specimen through DCPD measurements, the model can be used to indirectly calculate out-of-plane resistivity.

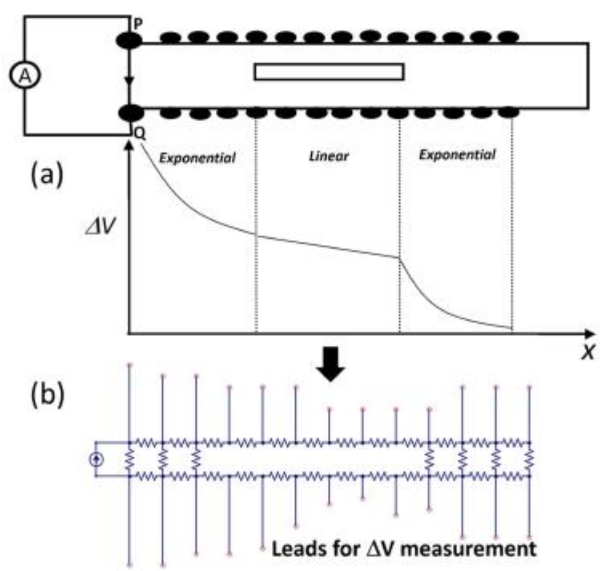


Figure 5: Ladder network of resistors[2]

5. Existing Test Equipment

Current market has specialized test instruments that implement the 4-probe technique. For instance, Jandel's general purpose four-point probe system measures electrical resistivity of metal layers, ingots and wafers up to 250 mm – 300 mm deep[3]. However, these instruments cannot be directly implemented since multiple probes will be required for crack detection requiring elaborate setup. National Instruments provides a range of DAQ systems however popular DAQ boards such as the NI-9205 or NI 9263 have limited differential analog input channels, they are expensive and require proprietary software for data acquisition[4]. Therefore, a custom test instrument was designed with a series of 4-probe measurement setup combined with an open source DAQ systems as detailed in the following section.

6. Proposed Design

A measurement device to implement DCPD as a portable and flexible technique to detect cracks was designed by connecting a pair of probe arrays to a DAQ system.

PROBING TOOL – An array of Mill-Max model 826 double row spring loaded connectors with 10 pins/row with a uniform spacing of 2.54mm between pins were selected. The accuracy of the DCPD technique was highly sensitive to material surface finish, contact force and contact resistance. Therefore, to accommodate for surface unevenness and decrease contact resistance, the 826 model connectors made of copper alloy with nickel and gold plating and with a stroke of 2.3mm were selected. Other specifications of the connectors can be found in the data sheet attached in Appendix A1. The probes were mounted on a handheld fixture that contained alignment magnets. A schematic of the probing tool assembly is shown in Figure 6(a).

DAQ SYSTEM – An Arduino Mega containing 15 analog channels was used. The Mega was incapable of reading differential analog inputs. Therefore, relays were used to switch ground reference and record potential difference one pair at a time. This circuit design for a five-probe measurement setup is illustrated in Figure 6(b).

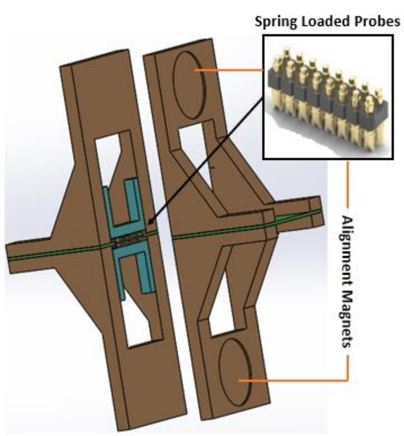


Figure 6 (a): Probing Tool Assembly

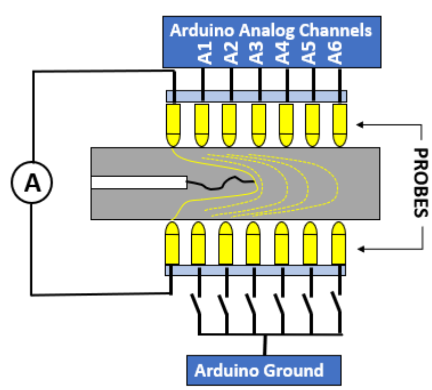


Figure 6 (b): Circuit Schematic

6.1 DAQ System Resolution and Accuracy

Using a 100mA current source, potential difference in 4mm-6mm thick composite materials typically drop from 70mV – 60mV to 5mV – 1mV across 25mm from the current source. Therefore, a DAQ system with a resolution of 0.5mV/bit or lower was required.

The ATMega 2560 chip was a 10-bit analog to digital converter that measured up to 5V of input signal.[5] It converted a 0-5V reading into a 0-1023 bit reading which gave it a resolution of 0.0049V/bit or 4.9mV/bit. The resolution of the Mega could be increased in two ways. One, the internal 1.1V reference could be used instead of the Mega board's default 5V reference. This increased its resolution to:

$$\frac{1.1\text{V}}{1023} \approx \frac{0.98\text{ mV}}{\text{bit}}$$

Although this was lower than the expected resolution, it was a simple INTERNAL1V1 command line to switch default voltage reference to 1.1V. Therefore, this method was used to validate the DAQ system in the prototype phase. In the final product, however, a higher resolution is intended to be achieved using operational amplifiers. Op-amps amplified voltage signals with a set gain and therefore increased the resolution of the DAQ by the same proportion. For instance, an op-amp with a gain of 500 connected to an Arduino Mega with default 5V reference provided a resolution of:

$$\pm\frac{5\text{V}}{1023\text{ bits}}\left(\frac{1}{500}\right) = 9.8(10^{-6})\text{V}$$

Implementing op-amps with the potential drop measurement device was extensively worked on during the prototype phase during which key limitations and complexities were realized:

i. Saturation – Increase in resolution decreased range of measurement of the DAQ system. In other words, a 5V ADC with a 500 gain could now measure only 0-10mV. Any signal higher than 10mV would rail the op-amp to saturation. Therefore, a voltage divider circuit as shown in Figure 7 was needed to measure voltages greater than 5V.

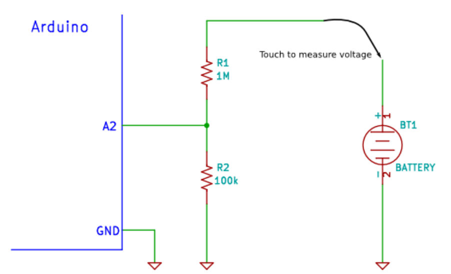


Figure 7: Voltage divider circuit[9]

ii. Signal Processing and noise – A buffer (unity gain op-amp) is to be added to prevent loading of the input signal to the non-inverting amplifier. Next a low pass filer to eliminate high frequency DC noise is connected in series with the output of the non-inverting amplifier. The resistors used in the RC filter interfered with the gain ratio of the non-inverting amplifier which gave undesirable output signals with op-amps implemented in the prototype phase.

This circuit schematic for one measurement probe using buffer, low pass filter, non-inverting amplifier configuration is shown in Figure 8.

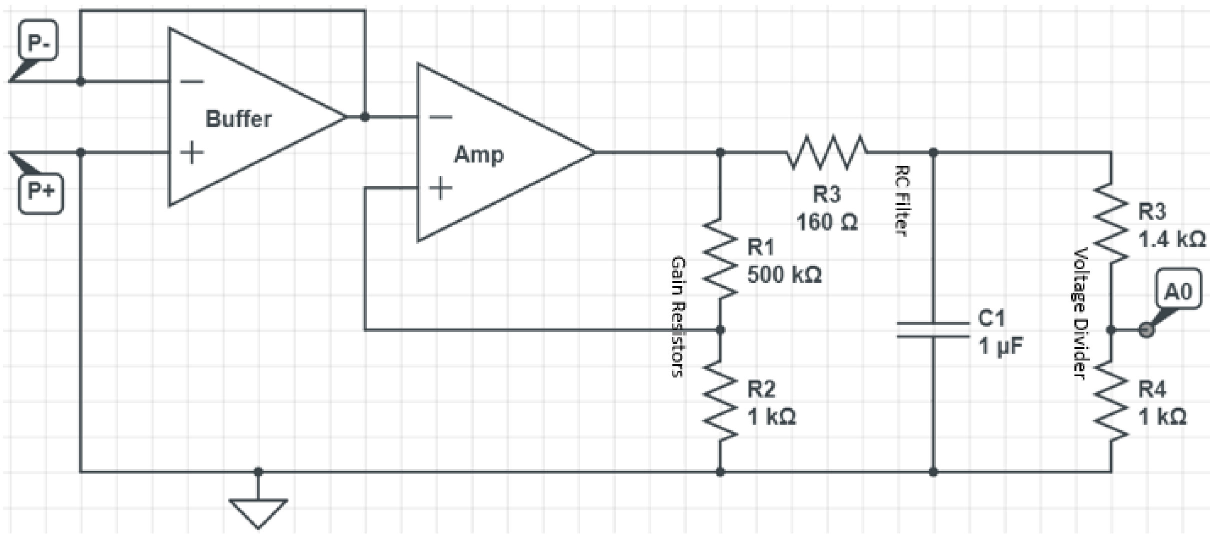


Figure 8: Signal amplification circuit [6]

The above circuit would be used to amplify each voltage signal. The amplified output is calculated using the following equations:

Amplifier Gain (non-inverting configuration)[7]:

$$G = \frac{R_1}{R_2} + 1 = \frac{500k}{100} + 1 = 501$$

A DC noise frequency of 1kHz was assumed, a 1μF capacitance was selected to calculate a resistance value[10]:

$$\frac{1}{2\pi RC} = 1\text{kHz} \implies R = 160\Omega$$

The resistors in the voltage divider circuit were chosen depending on the maximum voltage supply to the amplifier, because amplifiers saturated at their power supply voltage. Assuming a 12V supply, the resistance values depicted in Figure 8 were chosen. Therefore, if the amplifier produced a maximum output voltage of 12V, the voltage divider stepped the output signal down to:

$$\left(\frac{R_4}{R_3 + R_4}\right)V = \frac{1}{1 + 1.4}(12\text{ V}) = 5\text{V}$$

According to the datasheet[5], the accuracy of the ADC was ±2 LSB which was ±9.7mV in a 5V setting. Using the internal 1.1V reference the accuracy was increased to ±2.14mV. Further, using a potentiometer the accuracy of the Mega board was measured using a multimeter. The accuracy was found to be within 2% of the true value (Appendix A2).

6.2 Data Analysis

MATLAB was used to further analyze trend in potential drop data obtained from the DAQ system. The code was developed to identify and discriminate an exponential and linear decay in potential drop across length of the specimen. Since the spring loaded connectors were

equally spaced with a pitch of 2.54mm (0.1in), a position vector relative to the first pin with a 2.54mm (0.1in) increment was plotted against measured potential difference.
A linear and exponential trend was differentiated by calculating the second derivative of the function. The rate of change of gradient of a linear function was essentially zero whereas that of an exponential function was not (i.e. it was a multiple of the original function). The MATLAB code exploited this concept by numerically calculating the second derivative of the given set of data points. Since no data set was ideal, the second derivative was expected to be non-zero. Therefore, a range of zero approximation had to be defined to make the code realistic:

$$dm = 0 \Rightarrow -0.1 < dm < 0.1, where\ dm\ equals\ second\ derivative\ of\ a\ linear\ function$$

The point of transition between a linear and exponential trend indicated crack tip, and the length of the crack was equal to the length of the linear fit region.
First, the code was tested against set data points derived from a two-piece function:

$$y = e^{-3(x-3)} + 1 ; 0 \leq x \leq 4.3$$
$$y = -x + 5.3 ; x > 5.3$$

Online graphing calculator was used to develop these functions and intervals through trial and error. A set of 6 data points listed in Table 1 were chosen close to the defined functions:

X	Y
3.7	1.12
4.0	1.05
4.3	1.02
4.4	0.9
4.5	0.8
4.6	0.7

Table 1: Data Points

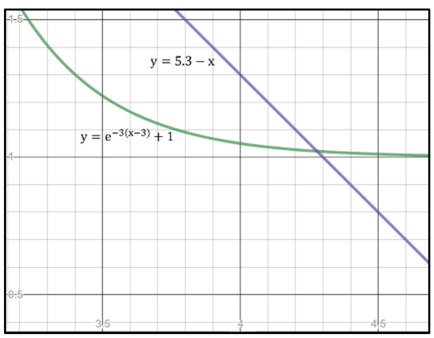


Figure 9(a): Functions plotted on Desmos to select data points

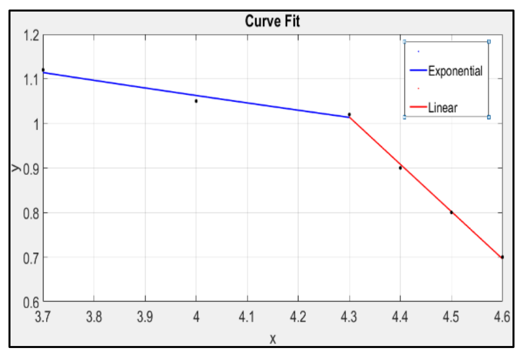


Figure 9(b): MATLAB plot of piecewise function

As seen from Figure 9(b), the MATLAB code successfully identified the point of transition. On an important note, the transition point identification of the code crucially depended on suitable selection of *dm* value. In the above simulation a *dm* range of ±0.1 was chosen. A very high dm range would apply a linear fit to the entire data point, while a small range would apply a global exponential curve fit! Consequently, erroneous transition point estimations were obtained from inappropriate values of dm and from data sets involving large standard deviations. The complete MATLAB code can be found in Appendix A5.

6.3 Manufacturing and Assembly of Device

The probe fixture parts were 3D printed and the spring-loaded connectors attached using high strength adhesive. Spring loaded connectors with solder cups were used for ease of wire attachment. A DAQ box to contain and protect electrical components and connections was

designed as shown in Figure 10(b). The alignment magnets had an effective magnetic field range of 10-12mm therefore were suited to measure specimens 4-8mm thick.

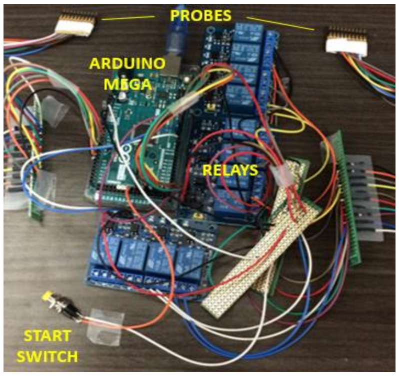


Figure 10 (a): Circuit Assembly

Figure 10 (b): 3D printed probe fixture and DAQ box

6.4 Testing & Results

A SiC/SiC notched specimen was loaded to propagate crack along the length of a 75mm x 13.5mm x 4.5mm specimen. Silver leads were applied to the notched edge to enable current supply. The tested specimen was placed between the negative and positive probes. A constant current source to provide stable 100mA DC current was connected to the first pair of leads. A starter switch was pressed, and data acquisition was initiated. The first relay switch closed the ground circuit and connected the first negative terminal probe to Arduino ground. A delay of 3 seconds was given to stabilize the current before measuring voltage drop and 10 data points were recorded for each pin every 0.5s. It was important to have only one probe circuit closed at a given time to prevent 'leak' in potential drop. Then the analog signal was read, and the first circuit was opened. Simultaneously the next relay switch closed the second circuit and the same process was repeated to obtain 10 data points. A detailed Arduino script for timing the relay switch and the data acquisition process can be found in Appendix A4. Data was collected from the DAQ system and analyzed using the MATLAB code developed. The estimation of crack length measured using the prototype device is plotted in Figure 11:

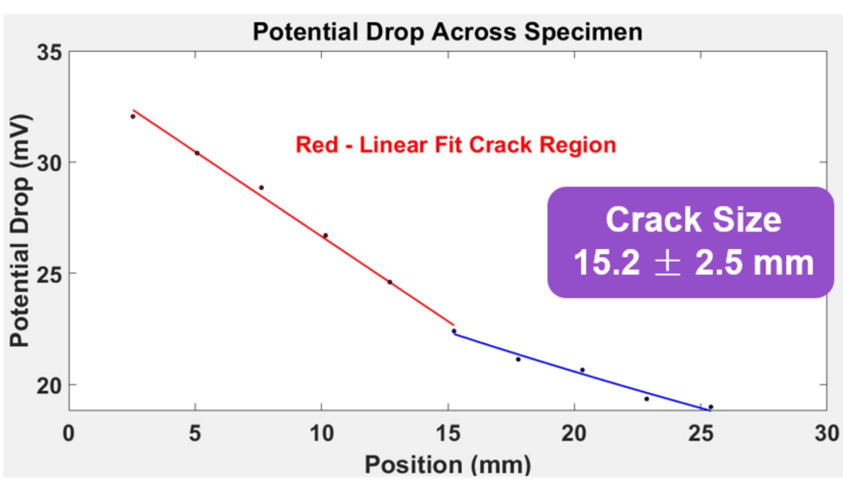


Figure 11: Potential drop across SiC/SiC specimen

It is important to note that the position was measured relative to pin one of the probe. In this case, probe one was placed 2.5mm (0.1'') below the notch tip, therefore the plot indicated a 15 cm long crack from the notch tip. The uncertainty in the estimation was equal to pin spacing of the probes 2.5mm (0.1''). Crack length in the same SiC/SiC specimen was calculated through visual examination as shown in Figure 12. The crack length estimate using the DCPD prototype device were consistent with microscopic observation.

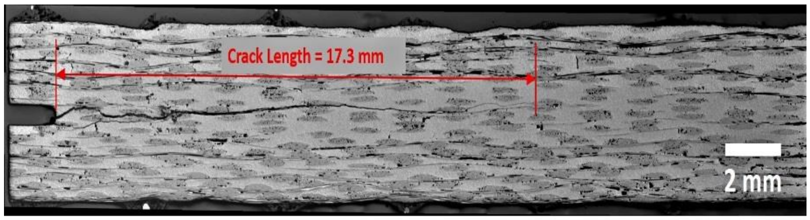


Figure 12: Crack size measured using microscope

7. Future Work

The DCPD prototype device showed positive results to implement it as a non-destructive technique for crack detection. The following improvisations were realized for developing the prototype into a full scale measurement device:

1. **Test multiple composite materials – The technique needs to be validated for other composite materials such as C/C-SiC materials and composites with different surface finish, fiber orientations and manufactured processes.**
2. **Implement op-amps – As detailed earlier, plenty of rudimentary research has been done on implementing op-amp circuits to this measurement device. Practical issues such as resistance overlap and saturation need to be controlled to successfully implement op-amps and greatly increase the resolution of the DAQ system.**
3. **Dependence on position of current source – Having the current source placed on a notched region forced current to pass in on direction. However, in real applications, the current probes may have to be placed between arbitrary points on a material's surface. In that case, current flow between terminals may have multiple spreading directions, thereby decreasing strength (magnitude) of potential drop per path/direction.**
4. **Identifying various types of defects – The prototype developed in this project effectively measured 2D cracks, however, current spreading and the trend in potential drop data for current propagation across 1D cracks, surface cracks, partial cracks (fractured matrix but intact fibers) and cracks in different orientations need to be examined.**
5. **Implement constant current source from Arduino – Op-amps can also be used in advance configurations to convert a constant DC voltage to a constant current source.**

This would eliminate the need for current leads to be applied as the power supply can be integrated into the DAQ system.

6. **Arduino Multiplexer[7] – Data acquisition from all 10 channels of the measurement device could be recorded simultaneously if the number of available analog channels in the DAQ could be expanded. Multiplex shields for Arduino serve that purpose and should be considered to increase time efficiency.**
7. **Refine fixture design – A flexible mechanism to adjust probe alignment, replace pins and re-solder cables periodically should be integrated in the fixture design.**
8. **Filter noise and outliers – Transition point calculation by the MATLAB code developed was highly susceptible to error in the presence of noise and poor-quality data. The code could be refined by normalizing the first and second derivative calculated numerically, thereby making it independent of potential drop magnitude and unit change.**
9. **Spatial plot of test data - In real applications, one is more interested in detecting delamination and cracks over a surface (like a flat plate). The present MALTAB code can be further developed to create a plot from input specimen dimensions and test data from different probe orientations to create a 2D plot of the defects detected over different locations on the surface.**
10. **Calibration – To detect delamination, only the trend in potential drop data was of interest. However, if this data was to be used further to model ladder resistor network and calculate in-plane resistivity, a method to calibrate the device needed to be developed. This could be done by comparing absolute voltage measurements against another device or by maintaining control specimen whose resistivity values are known with high accuracy.**

8. Conclusion

Non-Destructive Testing is extensively used in industry to monitor and predict failure of mechanical components. NDTs are crucial especially for brittle materials such as CMCs in which fracture with no apparent plastic deformation occurs. Electrical properties of CMCs can be measured using the DCPD method. Moreover, analyzing potential drop trends have shown to be capable of estimating location and size of delamination. This project has demonstrated the feasibility of applying this novel concept of damage detection through analysis of electrical properties in composite materials. Preliminary results of this project have shown consistent results with other established techniques of crack detection. The project has also provided lots of technical and conceptual insights into issues that need to be addressed to develop the DCPD technique in to an effective NDT measurement device.

9. References

[1] G. N. Morscher, C. Smith, E. Maillet, C. Baker and R. Mansour, American Ceramic Society Bulletin, September 2014, Vol. 93, No. 7.

[2] Y. P. Singh, R. Panakarajupally, M. J. Presby, and G. N. Morscher, Composites Part B: Engineering (Elsevier Science, June 2019), Vol. 166, 722-730.

[3] Jandel Four Pint Probes, URL: http://www.jandel.co.uk/products/four-point-probes.html

[4] NI-9205 C Series Voltage Input Module, URL: http://www.ni.com/en-us/support/model.ni-9205.html

[5] Precise Voltage Measurement with the Arduino Board, URL: http://www.ni.com/en-us/support/model.ni-9205.html

[6] Circuit Lab, URL: https://www.circuitlab.com/editor/#?id=7pq5wm

[7] Multiplexer, URL: https://www.egr.msu.edu/classes/ece480/capstone/spring13/group08/documents/dan.pdf

[8] Amplifier Gain, URL: https://www.allaboutcircuits.com/textbook/semiconductors/chpt-1/amplifier-gain/

[9] Measuring DC Voltage Using Arduino, URL: https://startingelectronics.org/articles/arduino/measuring-voltage-with-arduino/

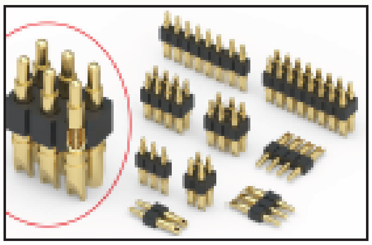
[10] RC Low Pass Filer, URL: http://www.ecircuitcenter.com/Calc/RCFilter_3_button/RC_Filter_3_button1.html

10. Appendix

A1: Probes Datasheet

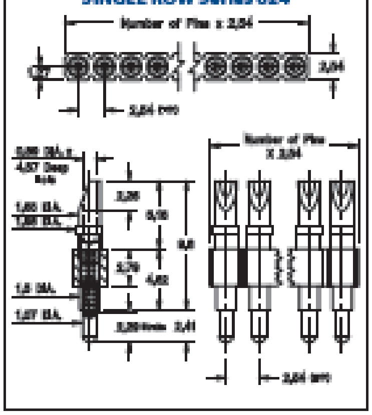
SPRING-LOADED CONNECTORS

SERIES 824 & 826 • 2,54 GRID SOLDERCUP HEADER • SINGLE AND DOUBLE ROW STRIPS

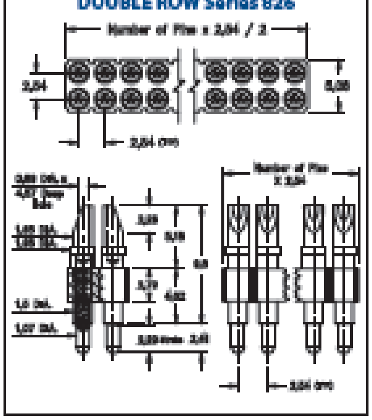


- Modular contacts for use on 2,54 grid, supplied in single and double row contact strips
- Precision-machined piston / base and gold-plated components assure up to 1,000,000 cycle life durability
- Pistons have a 1,14 mid. stroke and a 2,29 max. stroke
- Low resistance, high current contacts are rated at 2 amps continuous, 3 amps peak
- Insulators are high temperature thermoplastic
- Both 824 & 826 series strips have spring pins with wire termination soldercups. The soldercups are aligned to provide easy access for soldering up to size 24 AWG wires

SINGLE ROW Series 824



DOUBLE ROW Series 826



ORDERING INFORMATION

Single Row Series 824
824-22-0XX-00-005000 — Specify number of contacts 01-64
Double Row Series 826
826-22-0XX-00-005000 — Specify number of contacts 02-72

Technical Specifications

Materials:

Contact piston & base: Machined copper alloy plated 0,51µm gold over 2,54µm nickel
Spring: Beryllium copper-plated 0,25µm gold
Insulator: High temperature thermoplastic, rated UL94 V-0

Mechanical:

Spring force @ initial height: 25 grams
Spring force @ mid stroke (1,14): 60 grams
Durability: Up to 1,000,000 cycles

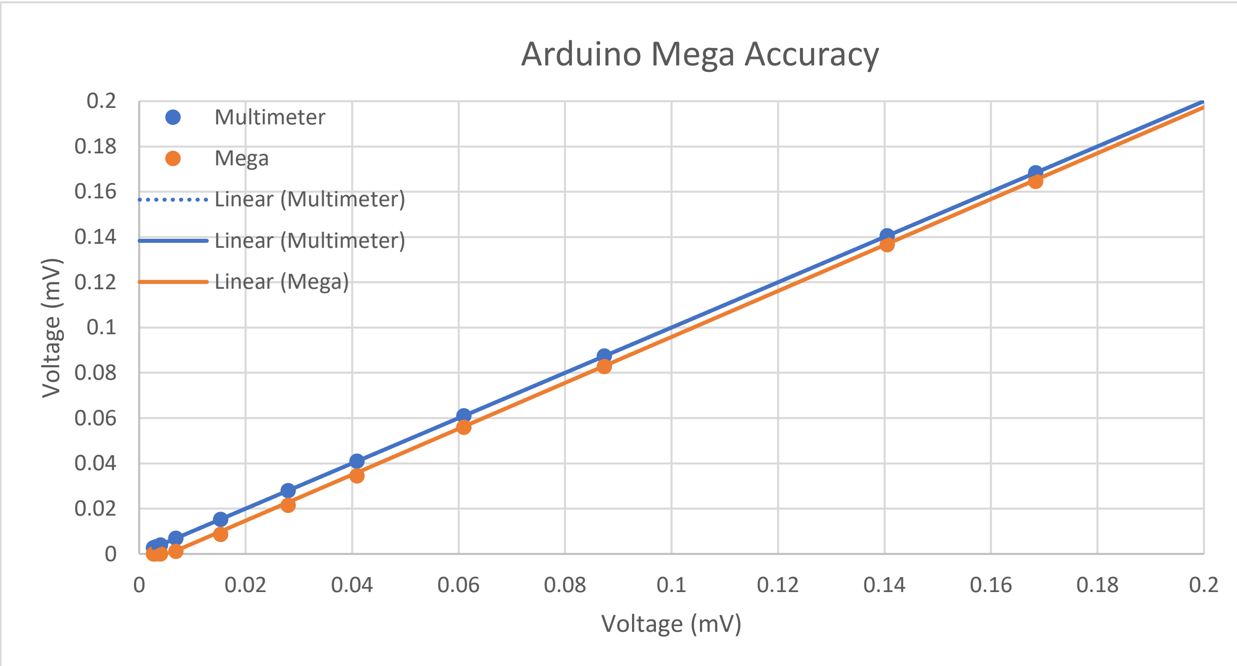
Electrical:

Voltage rating: 100Vrms/150Vdc
Current rating: 2A (continuous), 3A (peak) per contact
Contact resistance: 20mΩ max.
Insulation resistance: 10,000MΩ min.
Dielectric strength: 700Vrms min.
Capacitance: 1pF max.

RoHS-2 2011/65/EU

Mill-Max Mfg. Corp. • 190 Pine Hollow Road, P.O. Box 300, Oyster Bay, NY 11771 • 516-922-6000 • Fax: 516-922-9253 • www.mill-max.com

A2: Arduino Mega Accuracy Plot



Appendix A4: Arduino Script

```
void setup() {
analogReference(INTERNAL1V1);
pinMode(40,INPUT_PULLUP);
pinMode(2,OUTPUT);
digitalWrite(2,HIGH);
pinMode(4,OUTPUT);
digitalWrite(4,HIGH);
pinMode(6,OUTPUT);
digitalWrite(6,HIGH);
pinMode(8,OUTPUT);
digitalWrite(8,HIGH);
pinMode(10,OUTPUT);
digitalWrite(10,HIGH);
pinMode(12,OUTPUT);
digitalWrite(12,HIGH);
pinMode(22,OUTPUT);
digitalWrite(22,HIGH);
pinMode(26,OUTPUT);
digitalWrite(26,HIGH);
pinMode(30,OUTPUT);
digitalWrite(30,HIGH);
pinMode(34,OUTPUT);
digitalWrite(34,HIGH);

pinMode(A0,INPUT);
pinMode(A1,INPUT);
pinMode(A2,INPUT);
pinMode(A3,INPUT);
pinMode(A4,INPUT);
pinMode(A5,INPUT);
pinMode(A6,INPUT);
pinMode(A7,INPUT);
pinMode(A8,INPUT);
pinMode(A9,INPUT);

Serial.begin(9600);
}

void loop() {

if(digitalRead(40) == LOW)
```

```
{
  delay(9000);
   digitalWrite(2,LOW);               // Turns ON Relays 1
   for(int a=1;a<=9;a++){
    Serial.print(analogRead(A0)*1093.00/1023);
    Serial.print(",");
    delay(500);
   }
   Serial.println(analogRead(A0)*1093.00/1023);
   digitalWrite(2,HIGH);          // Turns Relay 1 Off
   digitalWrite(4,LOW);               // Turns ON Relays 2
   for(int b=1;b<=9;b++){
    Serial.print(analogRead(A1)*1093.00/1023);
    Serial.print(",");
    delay(500);
   }
   Serial.println(analogRead(A1)*1093.00/1023);
   digitalWrite(4,HIGH);          // Turns Relay 2 Off
   digitalWrite(6,LOW);               // Turns ON Relays 3
   for(int c=1;c<=9;c++){
   Serial.print(analogRead(A2)*1093.00/1023);
   Serial.print(",");
   delay(500);
   }
   Serial.println(analogRead(A2)*1093.00/1023);
   digitalWrite(6,HIGH);          // Turns Relay 3 Off
   digitalWrite(8,LOW);               // Turns ON Relays 4
   for(int d=1;d<=9;d++){
    Serial.print(analogRead(A3)*1093.00/1023);
    Serial.print(",");
    delay(500);
   }
   Serial.println(analogRead(A3)*1093.00/1023);
   digitalWrite(8,HIGH);          // Turns Relay 4 Off
   digitalWrite(10,LOW);               // Turns ON Relays 5
   for(int e=1;e<=9;e++){
    Serial.print(analogRead(A4)*1093.00/1023);
    Serial.print(",");
    delay(500);
   }
   Serial.println(analogRead(A4)*1093.00/1023);
   digitalWrite(10,HIGH);          // Turns Relay 5 Off
```

```
    digitalWrite(12,LOW);                  // Turns ON Relays 6
    for(int f=1;f<=9;f++){
     Serial.print(analogRead(A5)*1093.00/1023);
     Serial.print(",");
     delay(500);
    }
    Serial.println(analogRead(A5)*1093.00/1023);
    digitalWrite(12,HIGH);             // Turns Relay 6 Off
    digitalWrite(22,LOW);                  // Turns ON Relays 7
    for(int g=1;g<=9;g++){
     Serial.print(analogRead(A6)*1093.00/1023);
     Serial.print(",");
     delay(500);
    }
    Serial.println(analogRead(A6)*1093.00/1023);
    digitalWrite(22,HIGH);             // Turns 7 Relay Off
    digitalWrite(26,LOW);                  // Turns ON Relays 8
    for(int h=1;h<=9;h++){
     Serial.print(analogRead(A7)*1093.00/1023);
     Serial.print(",");
     delay(500);
    }
    Serial.println(analogRead(A7)*1093.00/1023);
    digitalWrite(26,HIGH);             // Turns Relay 8 Off
    digitalWrite(30,LOW);                  // Turns ON Relays 9
    for(int i=1;i<=9;i++){
     Serial.print(analogRead(A8)*1093.00/1023);
     Serial.print(",");
     delay(500);
    }
    Serial.println(analogRead(A8)*1093.00/1023);
    digitalWrite(30,HIGH);             // Turns Relay 9 Off
    digitalWrite(34,LOW);                  // Turns ON Relays 10
    for(int j=1;j<=9;j++){
     Serial.print(analogRead(A9)*1093.00/1023);
     Serial.print(",");
     delay(500);
    }
    Serial.println(analogRead(A9)*1093.00/1023);
    digitalWrite(34,HIGH);             // Turns Relay 10 Off
}
}
```

Appendix A5: MATLAB Code

```
clc
clear all

%Data Math Peicewise
x1 = [3.7 4.0 4.3 4.4 4.5 4.6];
y1 = [1.12 1.05 1.02 0.9 0.8 0.7];

%Data 1
% x1 = [0.1 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0];
% y1 = [32.8 28.9 25.8 24.9 21.1 20.7 20.5 18.4 18.6];

%Data 2
% x1 = [0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0];
% y1 = [29.2 28.4 26.7 24.6 24.6 21.1 20.6 20.5 20.5 18.8];

%Data 3
% x1 = [0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.9 1.0];
% y1 = [31.4 28.7 28.3 24.9 22.2 21.4 20.8 20.2 18.3];

%Data 4
% x1 = [0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0];
% y1 = [31.3 28.1 26.2 23.7 23.2 17.3 21.5 32.9 20.1 19.0];

%Data 5
% x1 = [0.1 0.4 0.6 0.7 0.8 0.9 1.0];
% y1 = [31.9 24.3 22.0 20.9 19.5 19.9 17.6];

%Data 6 Bad
% x1 = [0.1 0.2 0.3 0.4 0.5 0.7 0.8 0.9 1.0];
% y1 = [32,8 33.5 24.6 25.8 27.1 18.2 21.8 20.7 18.5];

%Data 7
% x1 = [0.1 0.4 0.5 0.7 0.8 0.9 1.0];
% y1 = [32.2 25.4 23.3 20.8 20.2 19.5 18.2];

%Data 8 (GOOD DATA)
% x1 = [0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0];
% y1 = [33.7 30.98 29.92 27.42 25.4 22.8 21.37 20.66 19.76 18.4];

%Data 9
% x1 = [0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1];
% y1 = [34.07 30.98 29.92 26.71 25.76 22.2 23.62 21.49 20.2 19.6];

%Data 10 (GOOD DATA)
% x = [0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1];
% x1 = 25.4*x;
% y1 = [32.05 30.4 28.85 26.7 24.6 22.4 21.13 20.65 19.35 18.99];

%Data 11 (Demo Specimen 1)
% x1 = [0.1 0.2 0.3 0.4 0.5 0.6 0.8 1];
% y1 = [29.79777778    28.01666667    27.06777778    25.40444444    24.69222222    21.25111111
```

```
        21.01222222             18.39888889]


for i=1:1:length(x1)-1
    X = [x1(i) x1(i+1)];
    Y = [y1(i) y1(i+1)];

    len = polyfit(X,Y,1);
    m(i) = len(1);
end
dm = diff(m)
%Br contains indices of all linear points
Br = find((diff(m)<0.1) & (diff(m)>-0.1));

%Assigning 1 - Linear and 0 - Expo points
for k =1:1:length(Br)
    B(Br(k)) = 1;
    B(Br(k)+1) = 1;
    B(Br(k)+2) = 1;
end
% Making sure end points do not form their own matrices
if B(1) ~= B(2)
    B(1) = B(2);
end
if B(length(B)) ~= B(length(B)-1)
    B(length(B)) = B(length(B)-1);
end
%Adding any trailing zeroes
B_len = length(B);
if B_len < length(x1)
c=1;
while c<=(length(x1)- B_len)
B(B_len+c)=0;
c=c+1; %c incremented to exit loop after 1st iteration
end
end

%Splitting Linear and exponential data sets
u =1; tA=1; tB=1; tC=1; tD=1;
for h=1:1:length(x1)
if u==1 && h ~=length(x1)
    if tA==1
        T(u) = B(h);
    end
    Ax(tA) = x1(h);
    Ay(tA) = y1(h);
    tA = tA+1;
    if B(h+1) ~= B(h)
%     Ax(tA) = x1(h+1);
%     Ay(tA) = y1(h+1);
    u = u+1;
    end

elseif u==2 && h ~=length(x1)
```

```
    if tB==1
        T(u) = B(h);
        Bx(tB) = Ax(length(Ax));
        By(tB) = Ay(length(Ay));
        tB = tB+1;
    end
    Bx(tB) = x1(h);
    By(tB) = y1(h);
    tB = tB+1;
    if B(h+1) ~= B(h)
%     Bx(tB) = x1(h+1);
%     By(tB) = y1(h+1);
        u = u+1;
    end

elseif u==3 && h ~=length(x1)
    if tC==1
        T(u) = B(h);
        Cx(tC) = Bx(length(Bx));
        Cy(tC) = By(length(By));
        tC = tC+1;
    end
    Cx(tC) = x1(h);
    Cy(tC) = y1(h);
    tC = tC+1;
    if B(h+1) ~= B(h)
%     Cx(tC) = x1(h+1);
%     Cy(tC) = y1(h+1);
        u = u+1;
    end

 elseif u==4 && h ~=length(x1)
    if tD==1
        T(u) = B(h);
        Dx(tD) = Cx(length(Cx));
        Dy(tD) = Cy(length(Cy));
        tD = tD+1;
    end
    Dx(tD) = x1(h);
    Dy(tD) = y1(h);
    tD = tD+1;
    if B(h+1) ~= B(h)
%    Dx(tD) = x1(h+1);
%    Dy(tD) = y1(h+1);
        u = u+1;
    end

elseif h == length(x1) && u == 1
    Ax(tA) = x1(h);
    Ay(tA) = y1(h);
elseif h == length(x1) && u == 2
    Bx(tB) = x1(h);
    By(tB) = y1(h);
elseif h == length(x1) && u == 3
```

```
    Cx(tC) = x1(h);
    Cy(tC) = y1(h);
elseif h == length(x1) && u == 4
    Dx(tD) = x1(h);
    Dy(tD) = y1(h);
end
end

% Curve Fitting split data sets

if length(T)>=1 && T(1)==1
    Afit = fit(Ax',Ay','poly1')
elseif length(T)>=1 && T(1)==0
    Afit = fit(Ax',Ay','exp1')
end

if length(T)>=2 && T(2)==1
    Bfit = fit(Bx',By','poly1')
elseif length(T)>=2 && T(2)==0
    Bfit = fit(Bx',By','exp1')
end

if length(T)>=3 && T(3)==1
    Cfit = fit(Cx',Cy','poly1')
elseif  length(T)>=3 && T(3)==0
    Cfit = fit(Cx',Cy','exp1')
end

if length(T)>=4 && T(4)==1
    Dfit = fit(Dx',Dy','poly1')
elseif  length(T)>=4 && T(4)==0
    Dfit = fit(Dx',Dy','exp1')
end

% Plotting Data Sets

if length(T) == 1
    if T(1) == 1
    pA=plot(Afit,Ax,Ay,'r-'); set(pA,'Color','red'); set(pA,'LineWidth',2); hold on
    else
    pA=plot(Afit,'b-'); hold on
    end
    scatter(Ax,Ay,18,'MarkerEdgeColor',[0 0 0],'MarkerFaceColor',[0 0 0]);
    set(legend,'visible','off'); hold off
elseif length(T) == 2 %%Plot AB
    if T(1) == 1
    pA=plot(Afit,Ax,Ay); set(pA,'Color','red'); set(pA,'LineWidth',2); hold on
    else
    pA=plot(Afit,Ax,Ay); set(pA,'Color','blue'); set(pA,'LineWidth',2); hold on
    end
    if T(2) == 1
    pB=plot(Bfit,Bx,By); set(pB,'Color','red'); set(pB,'LineWidth',2); hold on
    else
    pB=plot(Bfit,Bx,By); set(pB,'Color','blue'); set(pB,'LineWidth',2); hold on
```

```
    end
    sA = scatter(Ax,Ay,18,'MarkerEdgeColor',[0 0 0],'MarkerFaceColor',[0 0 0]);
    sB = scatter(Bx,By,18,'MarkerEdgeColor',[0 0 0],'MarkerFaceColor',[0 0 0]);
    set(legend,'visible','off'); hold off
%     a = annotation('textbox',[.2 .5 .3 .3],'String','Possible
Crack','Color','red','FitBoxToText','on');
elseif length(T) == 3 %%Plot ABC
    if T(1) == 1
    pA=plot(Afit,Ax,Ay); set(pA,'Color','red'); set(pA,'LineWidth',2); hold on
    else
    pA=plot(Afit,Ax,Ay); set(pA,'Color','blue'); set(pA,'LineWidth',2); hold on
    end
    if T(2) == 1
    pB=plot(Bfit,Bx,By); set(pB,'Color','red'); set(pB,'LineWidth',2); hold on
    else
    pB=plot(Bfit,Bx,By); set(pB,'Color','blue'); set(pB,'LineWidth',2); hold on
    end
    if T(3) == 1
    pC=plot(Cfit,Cx,Cy); set(pC,'Color','red'); set(pC,'LineWidth',2); hold on
    else
    pC=plot(Cfit,Cx,Cy); set(pC,'Color','blue'); set(pC,'LineWidth',2); hold on
    end
    sA = scatter(Ax,Ay,18,'MarkerEdgeColor',[0 0 0],'MarkerFaceColor',[0 0 0]);
    sB = scatter(Bx,By,18,'MarkerEdgeColor',[0 0 0],'MarkerFaceColor',[0 0 0]);
    sC = scatter(Cx,Cy,18,'MarkerEdgeColor',[0 0 0],'MarkerFaceColor',[0 0 0]);
    set(legend,'visible','off'); hold off
elseif length(T) == 4 %%Plot ABCD
    if T(1) == 1
    pA=plot(Afit,Ax,Ay); set(pA,'Color','red'); set(pA,'LineWidth',2); hold on
    else
    pA=plot(Afit,Ax,Ay); set(pA,'Color','blue'); set(pA,'LineWidth',2); hold on
    end
    if T(2) == 1
    pB=plot(Bfit,Bx,By); set(pB,'Color','red'); set(pB,'LineWidth',2); hold on
    else
    pB=plot(Bfit,Bx,By); set(pB,'Color','blue'); set(pB,'LineWidth',2); hold on
    end
    if T(3) == 1
    pC=plot(Cfit,Cx,Cy); set(pC,'Color','red'); set(pC,'LineWidth',2); hold on
    else
    pC=plot(Cfit,Cx,Cy); set(pC,'Color','blue'); set(pC,'LineWidth',2); hold on
    end
    if T(4) == 1
    pD=plot(Dfit,Dx,Dy); set(pD,'Color','red'); set(pD,'LineWidth',2); hold on
    else
    pD=plot(Dfit,Dx,Dy); set(pD,'Color','blue'); set(pD,'LineWidth',2); hold on
    end
    sA = scatter(Ax,Ay,18,'MarkerEdgeColor',[0 0 0],'MarkerFaceColor',[0 0 0]);
    sB = scatter(Bx,By,18,'MarkerEdgeColor',[0 0 0],'MarkerFaceColor',[0 0 0]);
    sC = scatter(Cx,Cy,18,'MarkerEdgeColor',[0 0 0],'MarkerFaceColor',[0 0 0]);
    sD = scatter(Dx,Dy,18,'MarkerEdgeColor',[0 0 0],'MarkerFaceColor',[0 0 0]);
    set(legend,'visible','off'); hold off
end
title('Crack Detection)
```

```
xlabel('Position')
ylabel('Potential Drop')
text(Ax(2),Ay(length(Ay)),'Red - Linear Fit Crack Region','Color','red','FontSize',14)
% text(Ax(2),Ay(length(Ay)),'Blue - Exp Fit','Color','blue','FontSize',14)
legend('','Exponential','','Linear')
grid on
```

Published with MATLAB® R2018a