calculate spring properties

Callback function to calculate spring properties

```
    function calculateSpring(endType, material, units, wireDiameter,
outerDiameter, freeLength, solidLength, fMin, fMax, peenedStatus)
```

Quality control

Check if end type and material are selected

```
        if isempty(endType) && isempty(material)
            errordlg('Please select End Type and Material', 'Error', 'modal');
            return;  % Exit the function if not selected
        elseif isempty(endType)
            errordlg('Please select End Type', 'Error', 'modal');
            return;  % Exit the function if not selected
        elseif isempty(material)
            errordlg('Please select Material', 'Error', 'modal');
            return;  % Exit the function if not selected
        end

         % Verification check for numeric values
    if ~isnumeric(wireDiameter) || ~isnumeric(outerDiameter) ||
~isnumeric(freeLength) || ~isnumeric(solidLength)
        errordlg('Please enter numeric values for diameters and lengths',
'Error', 'modal');
        return;
    end

    % Wire diameter verification
    if wireDiameter < 0
        errordlg('Invalid wire diameter. Please enter a positive value.',
'Error', 'modal');
        return;
    elseif wireDiameter == 0
        errordlg('Invalid wire diameter. Please enter a positive, non-zero
value.', 'Error', 'modal');
        return;
    end

    % Outer diameter verification
    if outerDiameter < 0
        errordlg('Invalid outer diameter. Please enter a positive value.',
'Error', 'modal');
        return;
    elseif outerDiameter == 0
        errordlg('Invalid outer diameter. Please enter a positive, non-zero
value.', 'Error', 'modal');
        return;
    end
    % Free length verification
    if freeLength <= solidLength || freeLength < 0
        errordlg('Invalid Free Length. Please enter a positive value greater
than Solid Length.', 'Error', 'modal');
        return;
    elseif freeLength == 0
        errordlg('Invalid Free Length. Please enter a non-zero, positive
value greater than Solid Length', 'Error', 'modal');
        return;
    end

    % Solid length verification
    if solidLength < 0
        errordlg('Invalid Solid Length. Please enter a positive value.',
'Error', 'modal');
        return;
    elseif solidLength == 0
        errordlg('Invalid Solid Length. Please enter a non-zero, positive
value.', 'Error', 'modal');
        return;
    end

    %peened or unpeened verification
    if fMin ~= 0 && isempty(peenedStatus)
        errordlg('If Fmin is not zero, peened or unpeened must be selected.',
'Error');
        return;
    elseif fMin == 0 && ~isempty(peenedStatus)
        errordlg('If Fmin is zero, neither peened nor unpeened should be
selected.', 'Error');
        return;
    end
```

```
Not enough input arguments.

Error in calculateSpring (line 6)
        if isempty(endType) && isempty(material)
```

Check if the units are in English or Metric

```
    if strcmp(units, 'English')
        % Convert to mm from inches and pounds based on Michael's test case
        convertToMetric_length = @(value) value * 25.4;  % from inches to mm
        convertToMetric_force = @(value) value * 4.44822; %from pound force
to newtons

        wireDiameter = convertToMetric_length(wireDiameter);
        solidLength = convertToMetric_length(solidLength);
        freeLength = convertToMetric_length(freeLength);
        outerDiameter = convertToMetric_length(outerDiameter);
        fMax = convertToMetric_force(fMax);
        fMin = convertToMetric_force(fMin);
        % Add any other conversions if needed
    end
 % Now user inputs are either in metric or remain unchanged
```

Call other functions to calculate and display results

```
        %use round to ensure the outcome is an integer
        totalCoils = round(calculateTotalCoils(endType, wireDiameter,
solidLength));
        activeCoils = round(calculateActiveCoils(endType, totalCoils));
        pitch = calculatePitch(activeCoils, freeLength, wireDiameter,
endType);
        springRate = calculateSpringRate(wireDiameter, outerDiameter,
activeCoils, material);
        force = calculateForce(freeLength, solidLength, springRate);
        force_FOS = calculateStaticFOS(material, wireDiameter, force,
outerDiameter);% fos calculated from force needed to compress the spring to
max length

        %calculate FOS - verify if static or inf life
        if fMin == 0
            fos = calculateStaticFOS(material, wireDiameter, fMax,
outerDiameter);
        else
            fos = calculateInfFOS(fMin, fMax, outerDiameter, wireDiameter,
peenedStatus, material);
        end

        % Display the results in a new figure
        displayResultsFigure(totalCoils, activeCoils, pitch, springRate,
force, fMin, fos, force_FOS, units);

    end
```

Published with MATLAB® R2023b