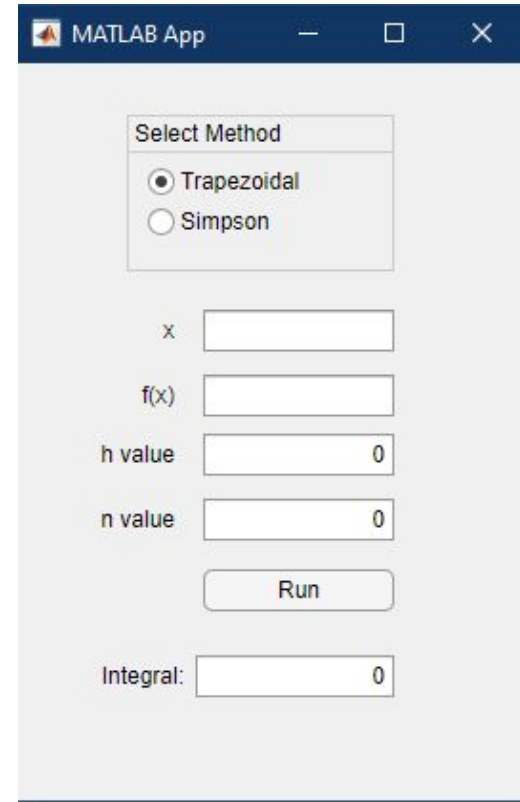


Simpson's and Trapezoidal Integration Rules

Giorgi Samushia and Evan Bilotta

GUI Interface and Inputs

- The GUI interface allows for either the Trapezoidal or Composite Simpson's method for evaluating the integral, along with input fields.
- These fields are: the points to be checked (x), the values at those points ($f(x)$), the increment (h), and the number of iterations (n). These are all passed to the system, and then passed to whichever function the user selects.



A screenshot of a MATLAB App window titled "MATLAB App". The interface is light gray and contains the following elements:

- Select Method:** A group box containing two radio buttons. The "Trapezoidal" button is selected (indicated by a black dot), and the "Simpson" button is unselected.
- x:** A text label followed by an empty rectangular input field.
- f(x):** A text label followed by an empty rectangular input field.
- h value:** A text label followed by a rectangular input field containing the number "0".
- n value:** A text label followed by a rectangular input field containing the number "0".
- Run:** A rectangular button with the text "Run".
- Integral:** A text label followed by a rectangular input field containing the number "0".

The Source Code

- The AppDesigner in Matlab allows for values to be attributes of an app object, so passing data from the frontend to the functions is done by calling the EditField member's value attribute from the app object.
- These fields are updated in realtime as users input data, select values, and push buttons.

```
% Properties that correspond to app components
properties (Access = public)
    UIFigure                matlab.ui.Figure
    IntegralEditField        matlab.ui.control.NumericEditField
    IntegralEditFieldLabel   matlab.ui.control.Label
    RunButton                matlab.ui.control.Button
    fxEditField              matlab.ui.control.EditField
    fxEditFieldLabel         matlab.ui.control.Label
    xEditField               matlab.ui.control.EditField
    xEditFieldLabel          matlab.ui.control.Label
    nvalueEditField          matlab.ui.control.NumericEditField
    nvalueEditFieldLabel     matlab.ui.control.Label
    hvalueEditField          matlab.ui.control.NumericEditField
    hvalueEditFieldLabel     matlab.ui.control.Label
    SelectMethodButtonGroup  matlab.ui.container.ButtonGroup
    SimpsonButton            matlab.ui.control.RadioButton
    TrapezoidalButton        matlab.ui.control.RadioButton
end
```

Source Code pt. 2

- First, the function retrieves the values of the data from the app object, then it assigns it to a variable for passing into the simpson's and trapezoidal functions.
- The values for x and $f(x)$ are vectors, so they needed to be scanned with the `textscan()` method from Matlab, which converts the inputted string to a cell list of size one, containing a vector in the first index.

```
methods (Access = private)
function res = func(app)
    n = app.nvalueEditField.Value;
    h = app.hvalueEditField.Value;
    t = textscan(app.xEditField.Value, '%f', 'Delimiter', ',');
    x = t{1};
    temp = textscan(app.fxEditField.Value, '%f', 'Delimiter', ',');
    fx = temp{1};
    res = simp(fx,x,n,h);
end
function res2 = func2(app)
    n = app.nvalueEditField.Value;
    h = app.hvalueEditField.Value;
    t = textscan(app.xEditField.Value, '%f', 'Delimiter', ',');
    x = t{1};
    temp = textscan(app.fxEditField.Value, '%f', 'Delimiter', ',');
    fx = temp{1};
    res2 = trap(fx,x,n,h);
end
end
```

Source Code pt. 3

- Result values are declared as variables that call the simpson and trapezoidal functions with the arguments defined in the previous slide. Both functions use Lagrange interpolation to calculate the values of the function at all points in the interval, including any missing step values.

```
function res = simp(fx,x,n,h)
    s=fx(1);
    size = length(x);
    for k = 1 : (n/2)-1
        fxx = Lagrange(x, fx, (x(1) + (2*k)*h));
        s = s + 2*fxx;
    end
    for k = 1 : n/2
        fxx = Lagrange(x, fx, (x(1) + ((2*k-1)*h)));
        s = s + 4*fxx;
    end
    s = s + fx(size);
    res = h/3 * s;
end
```

```
function res = trap(fx,x,n,h)
    s=fx(1);
    size = length(x);
    for i = 2 : n
        fxx = Lagrange(x, fx, x(1) + (i-1)*h);
        s = s + 2*fxx;
    end
    s = s + fx(size);
    res = h*s/2;
end
```

Executing via GUI

- The radio buttons in the GUI are part of a ButtonGroup called SelectMethod, which has an attribute called SelectedObject that can be compared with the app's button attribute of the same name and type. A simple if-else check can be done on this radio button to decide which function gets called to set the value of the result field.

```
% Callbacks that handle component events
methods (Access = private)

% Callback function: IntegralEditField, RunButton,
% SelectMethodButtonGroup
function RunButtonPushed(app, event)
    if(app.SelectMethodButtonGroup.SelectedObject == app.SimpsonButton)
        app.IntegralEditField.Value = func(app);

    else
        app.IntegralEditField.Value = func2(app);
    end
end
```