Quadrature Points Integration

Table of Contents

Problem

Integrate with Quadrature Points between 1 and 2 with 6points and compute Error

```
f = @(x) x./(x.^2+2).^3; % 1 to 2
xlims = [1 2];
N = 6; %Quadrature Points
trueSolution = 1/48;
```

Solution

Compute the Points between -1 and 1 using subroutine

```
[points, weights] = quadrature(N);
```

Use linear Interpolation to map -1 to 1 points to integration limits

```
ratio = 2/(xlims(2)-xlims(1)); % Ratio between
x = (xlims(1)+xlims(2))/2 + points / ratio; %linear interpolation/
mapping from -1 to 1 --> 1-2
```

Initialize Area and sum up function values multiplied by the weights

```
Area = 0;
for i = 1:length(x)
    Area = Area + weights(i) * f(x(i)) / ratio;
end
Error = abs((trueSolution - Area)/trueSolution * 100);
```

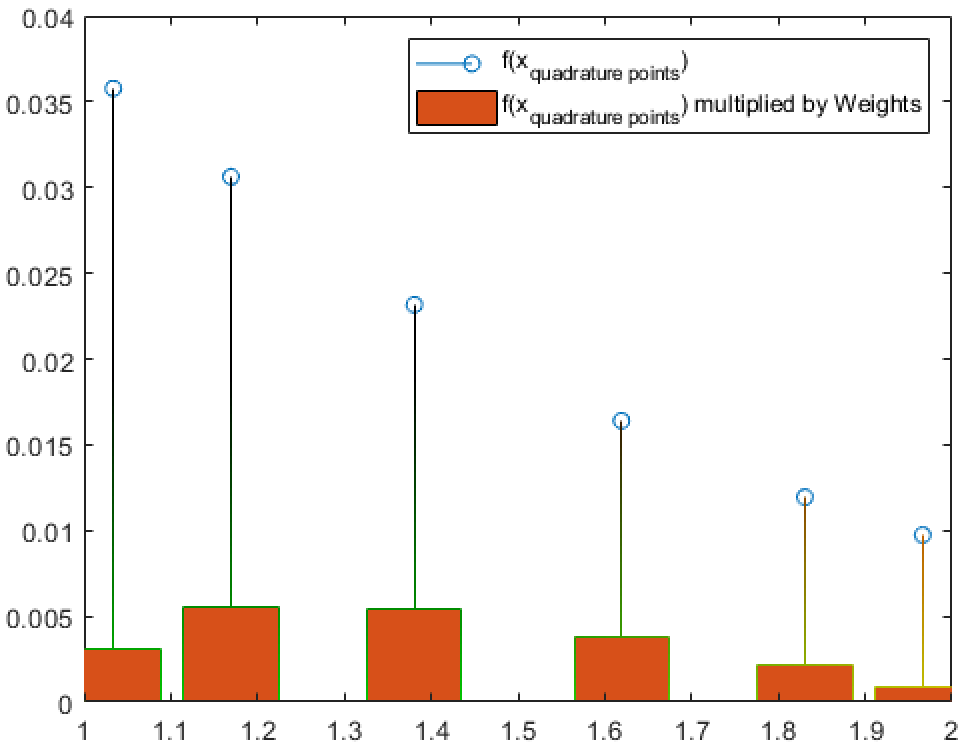
Display Value and Error

```
fprintf("Area = %.16f \nTrue Area = %.16f \nError = %.16e%%",Area,
 trueSolution, Error)
```

```
Area = 0.0208333333465953
True Area = 0.0208333333333333
Error = 6.3657612514589346e-08%
```

Visualization

```
stem(x,f(x));hold on
bar(x,f(x).*weights/ratio); hold off
xlim(xlims)
legend('f(x_{quadrature points})','f(x_{quadrature points}) multiplied
 by Weights')
```



Quadrature Points Generator Subroutine

```
type quadrature.m
```

```
function [points,weights] = quadrature(n)
%   QUADRATURE
%     quadrature(n) returns a quadrature table for a rule with n
%     integration points.  The first row of the table gives the
 quadrature
%     point location, and the second gives the quadrature weights.
if (n>=1e4)
    error('This will take too long!')
else
    u = 1:n-1;
    u = u./sqrt(4*u.^2 - 1);
    A = zeros(n);
    A(2:n+1:n*(n-1)) = u;
    A(n+1:n+1:n^2-1) = u;

    [v, points] = eig(A);
    [points, k] = sort(diag(points));
    points = points';
    weights = 2*v(1,k).^2;
end
```

Published with MATLAB® R2018b