Andrew J. Sommese, September 12, 2012

Problem 4.1: I am doing this quick and easy, i.e., I could do a procedure for T with N defined internal to the procedure.

```
> restart;
> N:=1;
```

$$N := 1 \tag{1}$$

```
> T := (f,h,N)-> (f(a)+2*sum(f(a+j*h),j=1..N-1)+f(a+N*h))*h/2;
```

$$T := (f, h, N) \rightarrow \frac{1}{2}\left(f(a) + 2\left(\sum_{j=1}^{N-1} f(a+jh)\right) + f(a+Nh)\right) h \tag{2}$$

```
> (4*T(f,h/2,2*N)-T(f,h,N))/3;
  simplify(%);
```

$$\frac{1}{3}\left(f(a) + 2 f\left(a + \frac{1}{2} h\right) + f(a+h)\right) h - \frac{1}{6}\left(f(a) + f(a+h)\right) h$$

$$\frac{1}{6} h f(a) + \frac{2}{3} h f\left(a + \frac{1}{2} h\right) + \frac{1}{6} h f(a+h) \tag{3}$$

note this is Simpson's rule   nodes at a, a+h, a+2h with weights h/6, 4*h/6, h/6 respectively

```
> (9*T(f,h/3,3*N)-T(f,h,N))/8;
  simplify(%);
```

$$\frac{3}{16}\left(f(a) + 2 f\left(a + \frac{1}{3} h\right) + 2 f\left(a + \frac{2}{3} h\right) + f(a+h)\right) h - \frac{1}{16}\left(f(a) + f(a+h)\right) h$$

$$\frac{1}{8} h f(a) + \frac{3}{8} h f\left(a + \frac{1}{3} h\right) + \frac{3}{8} h f\left(a + \frac{2}{3} h\right) + \frac{1}{8} h f(a+h) \tag{4}$$

note this is the Newton-Cotes rule for 4 nodes   a, a+h/3, a+2/3*h, a+h with weights h/8, 3/8*h, 3/8*h, h/8

```
> (16*T(f,h/4,4*N)-T(f,h,N))/15;
  simplify(%);
```

$$\frac{2}{15}\left(f(a) + 2 f\left(a + \frac{1}{4} h\right) + 2 f\left(a + \frac{1}{2} h\right) + 2 f\left(a + \frac{3}{4} h\right) + f(a+h)\right) h - \frac{1}{30}\left(f(a) + f(a + h)\right) h$$

$$\frac{1}{10} h f(a) + \frac{4}{15} h f\left(a + \frac{1}{4} h\right) + \frac{4}{15} h f\left(a + \frac{1}{2} h\right) + \frac{4}{15} h f\left(a + \frac{3}{4} h\right) + \frac{1}{10} h f(a+h) \tag{5}$$

This is a good rule, but certainly not one of the Newton-Cotes rules

Problem 4.2   There are several ways to do this. By definition one can integrate and choose the constant of integration to get the integral to be zero: this is easy by hand. I will use the generating function (no

```
> restart;
> int(1,x=0..1)
> taylor(w*exp(x*w)/(exp(w)-1),w,5);
```

$$1 + \left(x - \frac{1}{2}\right) w + \left(\frac{1}{12} + \frac{1}{2} x^2 - \frac{1}{2} x\right) w^2 + \left(\frac{1}{6} x^3 + \frac{1}{12} x - \frac{1}{4} x^2\right) w^3 + \left(-\frac{1}{720} + \frac{1}{24} x^4 + \frac{1}{24} x^2 - \frac{1}{12} x^3\right) w^4 + \mathrm{O}\left(w^5\right) \tag{6}$$

Problem 4.3

Let's now do the whole procedure for finding the points and weights of for Gaussian integration with an arbitrarily prescribed weight, w(t) on [a,b].

```
> restart:
```

```
with(LinearAlgebra):
Digits := 30;
a := -1;b:=1; #b:= Pi;
w := t ->1;#sqrt(t);# #sin(t)^2;
IP := proc( f,g ) evalf(int(f(t)*g(t)*w(t),t=a..b)) end;
```

$$Digits := 30$$
$$a := -1$$
$$b := 1$$
$$w := t \rightarrow 1$$
$$IP := \mathbf{proc}(f, g)\ \ evalf(int(f(t)*g(t)*w(t), t=a..b))\ \ \mathbf{end\ proc} \tag{7}$$

Note the inner product IP above

```
> n:=3;
```

$$n := 3 \tag{8}$$

```
> p := array(0..n);
> p[0] := 1:
> j:='j':k:='k':
> for j from 0 to n-1 do;
> vv := t*p[j]:
> for k from 0 to j do;
> vv := vv-IP(t*p[j],p[k])/IP(p[k],p[k])*p[k]:
> od:p[j+1]:=expand(vv):
> od:
```

$$p := array(0..3, [\ ]) \tag{9}$$

```
> for j from 0 to n do p[j]; od;
```

$$1$$
$$t + 2.92649536567162872787048949180\ 10^{-35}$$
$$t^2 + 1.17469536022685892148064162389\ 10^{-35}\ t - 0.333333333333333333333333333334$$
$$t^3 + 1.48884545369818515798320660658\ 10^{-35}\ t^2 - 0.599999999999999999999999999999\ t - 7.39158961349747032543756052997\ 10^{-36} \tag{10}$$

Here we compute the roots of the nth polynomial. There are fast solvers adapted to orthogonal polynomials --- if there is time we will discuss in class when we get to numerical linear algebra.

Lagrange basis functions for points indexed by i from 1 to N

```
> L :=  proc( N::integer, i::integer, v::Vector, x)
          description "Lagrange function";
  local j,T;
  T:=1:
  for j from 1 to N do
  if (j<>i) then T:= T*(x-v[j])/(v[i]-v[j])
  fi;
  od;
  T;
  end proc;
```

Computation of nodes x and weights

```
> for N from 1 to n do
  tempX:= Vector(N):
  x := Vector(N):
```

```
y:=Vector(N):
weights:=Vector(N):
tempX:= fsolve(p[N],t):
print("number of points" = N):
print('nodes'):
for j from 1 to N do
x[j] :=tempX[j]: print(x[j]):
od:
print('weights'):
for j from 1 to N do
weights[j]:=int(L(N,j,x,t),t=a..b):
print(weights[j]):
od:
od:
```

"number of points" = 1

nodes

$$\left(-2.926495365671628727870489491 80\ 10^{-35}\right)_1$$

weights

2

"number of points" = 2

nodes

-0.577350269189625764509148780503

0.577350269189625764509148780503

weights

0.999999999999999999999999999996

0.999999999999999999999999999996

"number of points" = 3

nodes

-0.774596669241483377035853079956

$-1.23193160224957838757292675500\ 10^{-35}$

0.774596669241483377035853079956

weights

0.555555555555555555555555555555

0.888888888888888888888888888880

0.555555555555555555555555555555 **(11)**

Let's verify the three point rule for polynomials of degree <= 5 (I went to 6 to show it is really exactly 5).

```
> k:='k':
j:='j':
for k from 0 to 6 do A:=int(t^k,t=a..b):
B:=0:
for j from 1 to 3 do
B:= B+weights[j]*x[j]^k:
od:
print(A-B):
od:
```

0.

0.

$3.\ 10^{-30}$

0.

$2.\ 10^{-30}$

0.

$$0.0457142857142857142857142857143 \quad \textbf{(12)}$$

Problem 4.4

```
> restart:
  with(LinearAlgebra):
  Digits := 30;
  a := 0;b:=1;
  w := t ->sqrt(t);
  IP := proc( f,g ) evalf(int(f(t)*g(t)*w(t),t=a..b)) end;
```

$$Digits := 30$$
$$a := 0$$
$$b := 1$$
$$w := t \rightarrow \sqrt{t}$$
$$IP := \mathbf{proc}(f, g) \;\; evalf(int(f(t) * g(t) * w(t), t = a..b)) \;\; \mathbf{end\ proc} \quad \textbf{(13)}$$

Note the inner product IP above

```
> n:=3;
```

$$n := 3 \quad \textbf{(14)}$$

```
> p := array(0..n);
> p[0] := 1:
> j:='j':k:='k':
> for j from 0 to n-1 do;
> vv := t*p[j]:
> for k from 0 to j do;
> vv := vv-IP(t*p[j],p[k])/IP(p[k],p[k])*p[k]:
> od:p[j+1]:=expand(vv):
> od:
```

$$p := array(0..3, [\;]) \quad \textbf{(15)}$$

```
> for j from 0 to n do p[j]; od;
```

$$1$$
$$t - 0.600000000000000000000000000000$$
$$t^2 - 1.11111111111111111111111111111\, t + 0.238095238095238095238095238096$$
$$t^3 - 1.61538461538461538461538461539\, t^2 + 0.734265734265734265734265734268\, t - 0.0815850815850815850815850815857 \quad \textbf{(16)}$$

Here we compute the roots of the nth polynomial. There are fast solvers adapted to orthogonal polynomials --- if there is time we will discuss in class when we get to numerical linear algebra.

Lagrange basis functions for points indexed by i from 1 to N

```
> L :=  proc( N::integer, i::integer, v::Vector, x)
          description "Lagrange function";
  local j,T;
  T:=1:
  for j from 1 to N do
  if (j<>i) then T:= T*(x-v[j])/(v[i]-v[j])
  fi;
  od;
  T;
  end proc;
```

$$L := \mathbf{proc}(N::integer, i::integer, v::Vector, x) \quad \textbf{(17)}$$
$$\mathbf{local}\ j, T;$$

description "Lagrange function";
$T := 1$; **for** j **to** N **do if** $j <> i$ **then** $T := T^* \, (x - v[j]) \, / \, (v[i] - v[j])$ **end if end do**; T
end proc

Computation of nodes x and weights

```
> for N from 1 to n do
  tempX:= Vector(N):
  x := Vector(N):
  y:=Vector(N):
  weights:=Vector(N):
  tempX:= fsolve(p[N],t):
  print("number of points" = N):
  print('nodes'):
  for j from 1 to N do
  x[j] :=tempX[j]: print(x[j]):
  od:
  print('weights'):
  for j from 1 to N do
  weights[j]:=int(L(N,j,x,t)*sqrt(t),t=a..b):
  print(weights[j]):
  od:
  od:
```

$$\text{"number of points"} = 1$$
$$nodes$$
$$0.6000000000000000000000000000000_{1}$$
$$weights$$
$$\frac{2}{3}$$
$$\text{"number of points"} = 2$$
$$nodes$$
$$0.2899491979256903022291517378 48$$
$$0.8211619131854208088819593732 62$$
$$weights$$
$$0.2775559982310616301347885316 15$$
$$0.3891106684356050365318781350 53$$
$$\text{"number of points"} = 3$$
$$nodes$$
$$0.1647102868965424215227963860 04$$
$$0.5498684992164435639085994625 54$$
$$0.9008058292716293991839887668 32$$
$$weights$$
$$0.1257826743288388479536492875 89$$
$$0.3076023676819127355061813001 05$$
$$0.2332816246559150832068360789 75 \tag{18}$$

Let's verify the three point rule for polynomials of degree <= 5 .

```
> k:='k':
  j:='j':
  for k from 0 to 5 do A:=int(t^k*sqrt(t),t=a..b):
  B:=0:
  for j from 1 to 3 do
  B:= B+weights[j]*x[j]^k:
  od:
```

```
print(A-B):
od:
```

$$-2.\ 10^{-30}$$
$$-2.\ 10^{-30}$$
$$-1.\ 10^{-30}$$
$$-2.\ 10^{-30}$$
$$-3.\ 10^{-30}$$
$$-5.\ 10^{-30} \qquad (19)$$