Archimedes Quadtrature in N Dimensions

```
> restart;
  with(plots):
```

Implement function S, as in the lecture slides:

```
> s:= proc(n,f,x,a,b,eps)
    #option trace;
    local value,ai,bi;
    # compute surplus and integrate over hat function in dim. n
    # (value is equiv. to D on the lecture slides)
    value:=archi(n-1,
              ( subs(x[n]=(a[n]+b[n])/2,f) -
                ( subs(x[n]=a[n],f) + subs(x[n]=b[n],f) )/2
              )
              *(b[n]-a[n])/2,
              x,a,b,eps);
    if abs(value) < eps
    then return value
    else
      # recursive calls to remainders S
      ai:=a: bi:=b:
      bi[n]:=(a[n]+b[n])/2:
      value:= value + s(n,f,x,ai,bi,eps,grid);
      bi[n]:=b[n]:
      ai[n]:=(a[n]+b[n])/2:
      value:= value + s(n,f,x,ai,bi,eps,grid)
    end if;
    return value;
  end proc;
```

```
s := proc(n, f, x, a, b, eps)
local value, ai, bi;
    value := archi(n - 1, 1 / 2*(subs(x[n] = 1 / 2*a[n] + 1 / 2*b[n], f)
        - 1 / 2*subs(x[n] = a[n], f) - 1 / 2*subs(x[n] = b[n], f))*(b[n] - a[n]), x, a, b,
        eps);
    if abs(value) < eps then return value
    else
        ai := a;
        bi := b;
        bi[n] := 1 / 2*a[n] + 1 / 2*b[n];
        value := value + s(n, f, x, ai, bi, eps, grid);
        bi[n] := b[n];
        ai[n] := 1 / 2*a[n] + 1 / 2*b[n];
        value := value + s(n, f, x, ai, bi, eps, grid)
    end if;
```

```
    return value
end proc
```

Archimedes quadratue:
- split function into trapezoid and surplus in dimension n
- return function value for dimension 0

```
> archi:= proc(n,f,x,a,b,eps)
    #option trace;
    global grid;
    if n=0
    then grid:= eval(grid) union {(a+b)/2};
         return f
    else evalf( archi(n-1,
                   ( subs(x[n]=a[n],f) + subs(x[n]=b[n],f) )
                   *(b[n]-a[n])/2,
                   x,a,b,eps)
              +s(n,f,x,a,b,eps)
             );
    end if;
  end proc;
```

```
archi := proc(n, f, x, a, b, eps)
global grid;
    if n = 0 then grid := eval(grid) union {1 / 2*a + 1 / 2*b}; return f
    else evalf(archi(n - 1,
        1 / 2*(subs(x[n] = a[n], f) + subs(x[n] = b[n], f))*(b[n] - a[n]), x, a, b, eps)
        + s(n, f, x, a, b, eps))
    end if
end proc
```

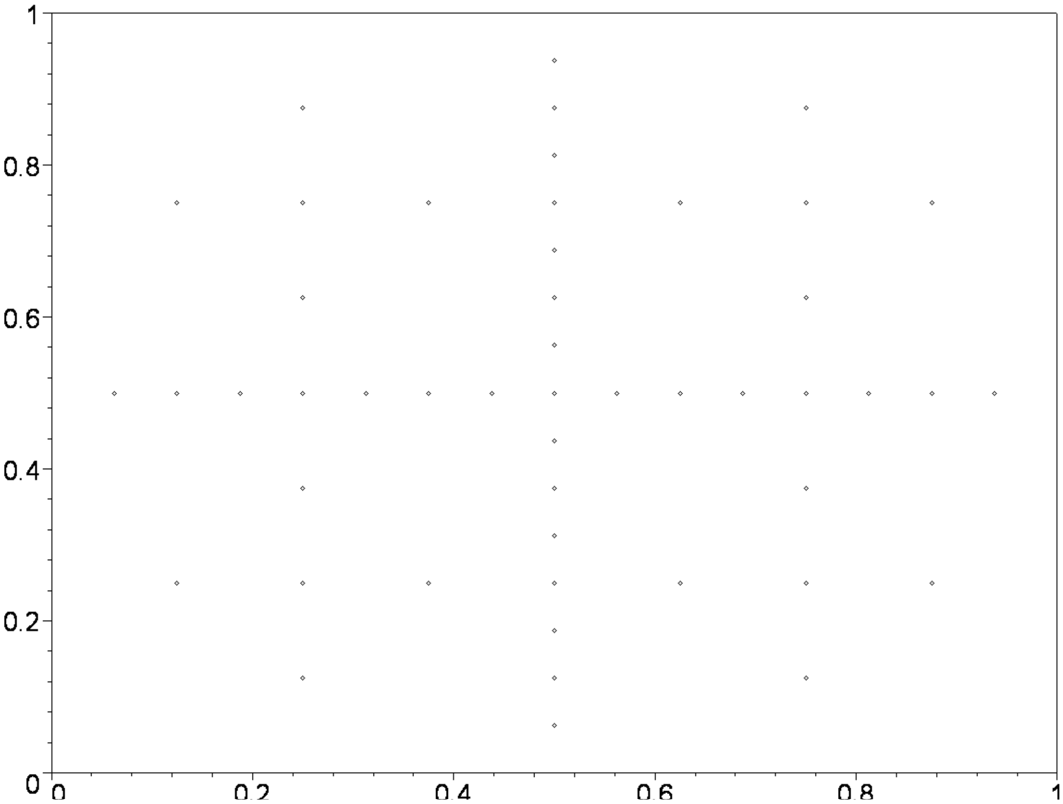
```
> f:= 36*x[1]*(1-x[1])*x[2]*(1-x[2]);
  #f:= sqrt(x[1])*(1-x[1])*sqrt(x[2])*(1-x[2]);
```

$$f := 36\, x_1\, (1 - x_1)\, x_2\, (1 - x_2)$$

```
> grid:={}:
> archi(2,f,x,[0,0],[1,1],1/200);
```

0.9843749999

```
> PLOT(POINTS(grid[],SYMBOL(DIAMOND)),AXESSTYLE(BOX),VIEW(0..1,0..
  1));
```



1
0.8
0.6
0.4
0.2
0
0
0.2
0.4
0.6
0.8
1

>

>

>