19.2 The angular frequency can be computed as $\omega_0 = 2\pi/360 = 0.017453$. Because the data are equispaced, the coefficients can be determined with Eqs. 19.14-19.16. The various summations required to set up the model can be determined as

t	Radiation	$\cos(\omega_0 t)$	$\sin(\omega_0 t)$	$y\cos(\omega_0 t)$	$y\sin(\omega_0 t)$
15	144	0.96593	0.25882	139.093	37.270
45	188	0.70711	0.70711	132.936	132.936
75	245	0.25882	0.96593	63.411	236.652
105	311	-0.25882	0.96593	-80.493	300.403
135	351	-0.70711	0.70711	-248.194	248.194
165	359	-0.96593	0.25882	-346.767	92.916
195	308	-0.96593	-0.25882	-297.505	-79.716
225	287	-0.70711	-0.70711	-202.940	-202.940
255	260	-0.25882	-0.96593	-67.293	-251.141
285	211	0.25882	-0.96593	54.611	-203.810
315	159	0.70711	-0.70711	112.430	-112.430
345	131	0.96593	-0.25882	126.536	-33.905
sum→	**2954**			**-614.175**	**164.429**

The coefficients can be determined as

$$A_0 = \frac{\Sigma y}{N} = \frac{2954}{12} = 246.1667$$

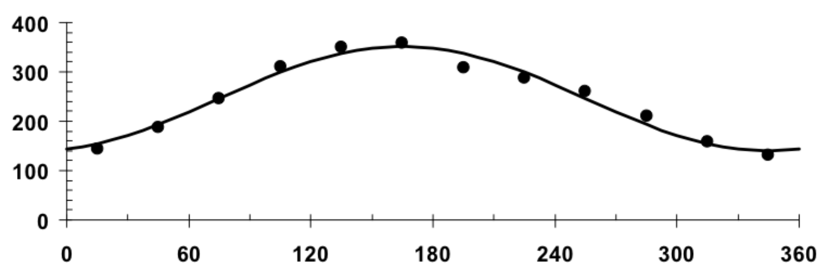
$$A_1 = \frac{2}{N}\Sigma y\cos(\omega_0 t) = \frac{2}{12}(-614.175) = -102.363$$

$$B_1 = \frac{2}{N}\Sigma y\sin(\omega_0 t) = \frac{2}{12}(164.429) = 27.4048$$

Therefore, the best-fit sinusoid is

$$R = 246.1667 - 102.363\cos(0.017453t) + 27.4048\sin(0.017453t)$$

The data and the model can be plotted as



The value for mid-August can be computed as

$$R = 246.1667 - 102.363\cos(0.017453(225)) + 27.4048\sin(0.017453(225)) = 299.1698$$

19.4 $a_0 = 0$

$$a_k = \frac{2}{T}\int_{-T/2}^{T/2} -2t\cos(k\omega_0 t)\,dt$$
$$= -\frac{4}{T}\left[\frac{1}{(k\omega_0)^2}\cos(k\omega_0 t) + \frac{t}{k\omega_0}\sin(k\omega_0 t)\right]_{-T/2}^{T/2}$$

$$b_k = \frac{2}{T}\int_{-T/2}^{T/2} -2t\sin(k\omega_0 t)\,dt$$
$$= -\frac{4}{T}\left[\frac{1}{(k\omega_0)^2}\sin(k\omega_0 t) - \frac{t}{k\omega_0}\cos(k\omega_0 t)\right]_{-T/2}^{T/2}$$

On the basis of these, all a's = 0. For k = odd,

$$b_k = \frac{2}{k\pi}$$

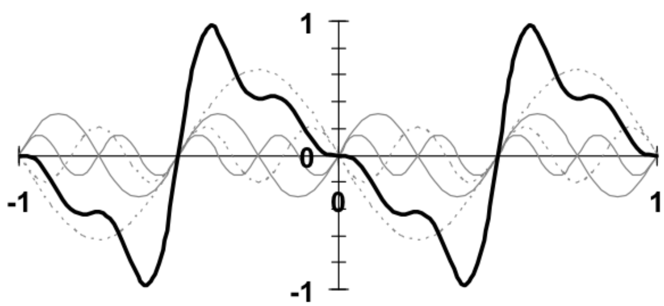
For k = even,

$$b_k = -\frac{2}{k\pi}$$

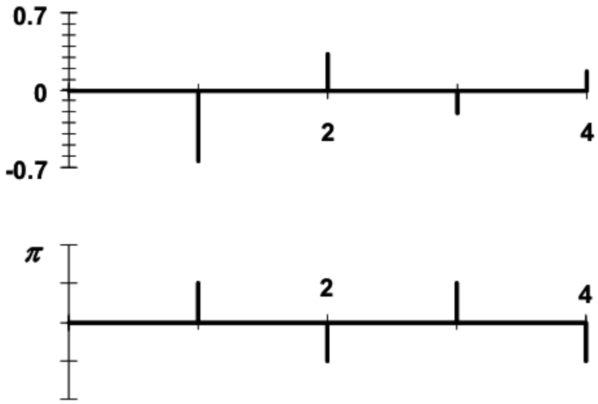
Therefore, the series is

$$f(t) = -\frac{2}{\pi}\sin(\omega_0 t) + \frac{1}{\pi}\sin(2\omega_0 t) - \frac{2}{3\pi}\sin(3\omega_0 t) + \frac{1}{2\pi}\sin(4\omega_0 t) + \cdots$$

The first 4 terms are plotted below along with the summation:



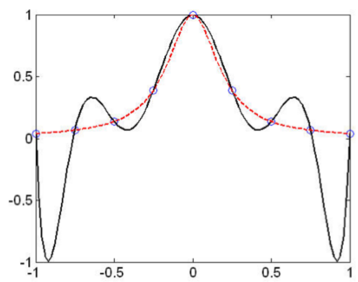
19.6



19.20 (a) Eighth-order polynomial:

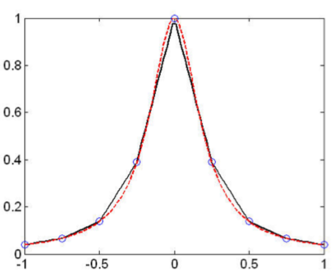
```
>> x=linspace(-1,1,9);

>> y=1./(1+25*x.^2);
>> p=polyfit(x,y,8);
>> xx=linspace(-1,1);
>> yy=polyval(p,xx);
>> yr=1./(1+25*xx.^2);
>> plot(x,y,'o',xx,yy,xx,yr,'--')
```



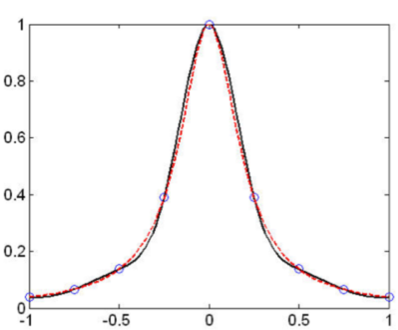
(b) linear spline:

```
>> x=linspace(-1,1,9);
>> y=1./(1+25*x.^2);
>> xx=linspace(-1,1);
>> yy=interp1(x,y,xx);
>> yr=1./(1+25*xx.^2);
>> plot(x,y,'o',xx,yy,xx,yr,'--')
```



(b) cubic spline:

```
>> x=linspace(-1,1,9);
>> y=1./(1+25*x.^2);
>> xx=linspace(-1,1);
>> yy=spline(x,y,xx);
>> yr=1./(1+25*xx.^2);
>> plot(x,y,'o',xx,yy,xx,yr,'--')
```



19.23 This problem is convenient to solve with MATLAB

(a) When we first try to fit a sixth-order interpolating polynomial, MATLAB displays the following error message

```
>> x=[0 100 200 400 600 800 1000];
>> y=[0 0.82436 1 .73576 .40601 .19915 .09158];
>> p=polyfit(x,y,6);
Warning: Polynomial is badly conditioned. Remove repeated data points
         or try centering and scaling as described in HELP POLYFIT.
> In polyfit at 79
```

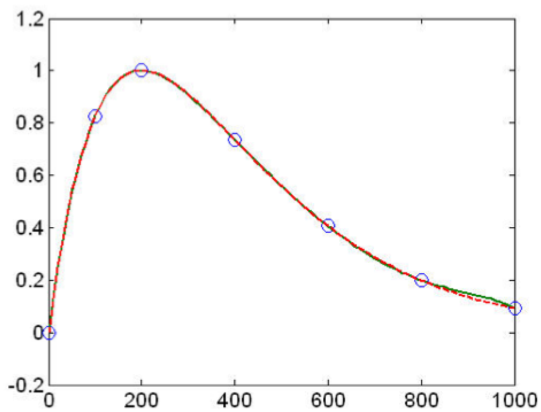
Therefore, we redo the calculation but centering and scaling the *x* values as shown,

```
>> xs=(x-500)/100;
>> p=polyfit(xs,y,6);
```

Now, there is no error message so we can proceed.

```
>> xx=linspace(0,1000);
>> yy=polyval(p,(xx-500)/100);
>> yc=xx/200.*exp(-xx/200+1);
>> plot(x,y,'o',xx,yy,xx,yc,'--')
```

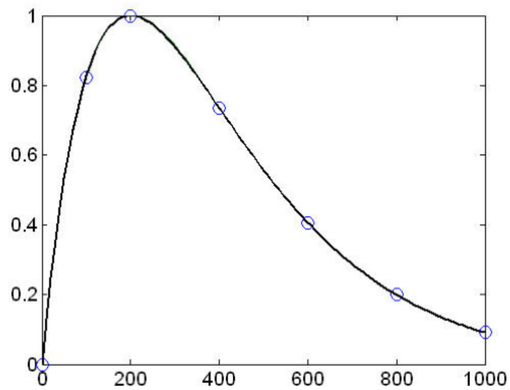
This results in a decent plot. Note that the interpolating polynomial (solid) and the original function (dashed) almost plot on top of each other:



(b) Cubic spline:

```
>> x=[0 100 200 400 600 800 1000];
>> y=[0 0.82436 1 .73576 .40601 .19915 .09158];
>> xx=linspace(0,1000);
>> yc=xx/200.*exp(-xx/200+1);
>> yy=spline(x,y,xx);
>> plot(x,y,'o',xx,yy,xx,yc,'--')
```

For this case, the fit is so good that the spline and the original function are indistinguishable.



(c) Cubic spline with clamped end conditions (zero slope):

```
>> x=[0 100 200 400 600 800 1000];
>> y=[0 0.82436 1 .73576 .40601 .19915 .09158];
>> ys=[0 y 0];
>> xx=linspace(0,1000);
>> yc=xx/200.*exp(-xx/200+1);
>> yy=spline(x,ys,xx);
>> plot(x,y,'o',xx,yy,xx,yc,'--')
```

For this case, the spline differs from the original function because the latter does not have zero end derivatives.