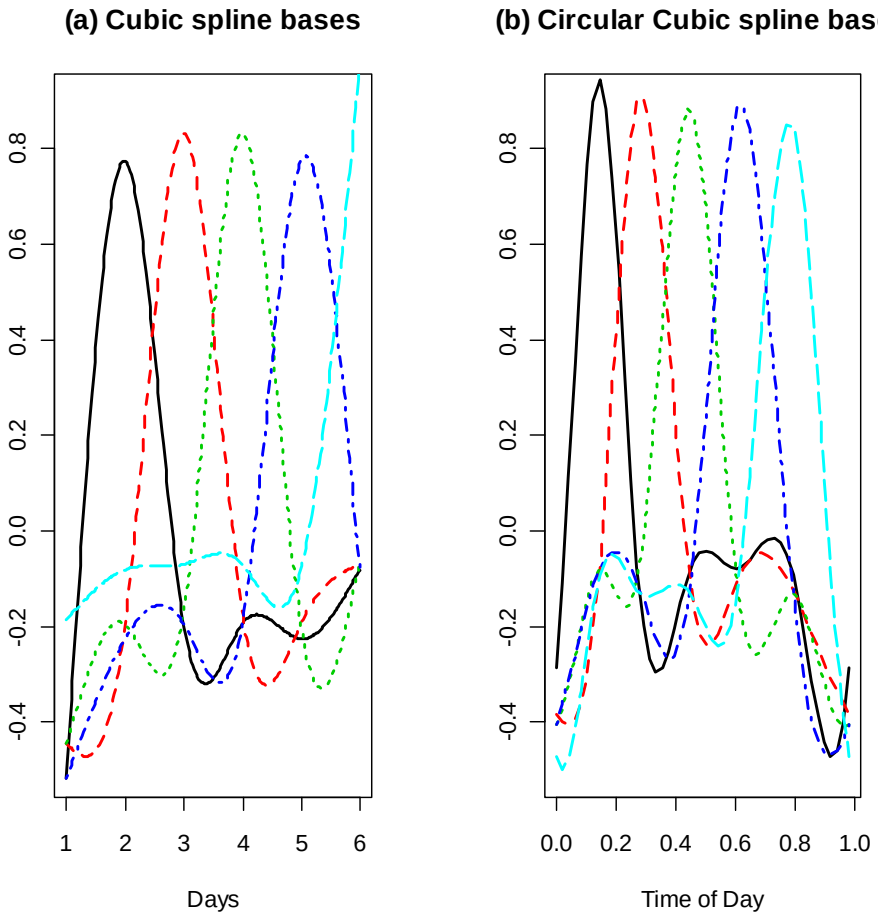
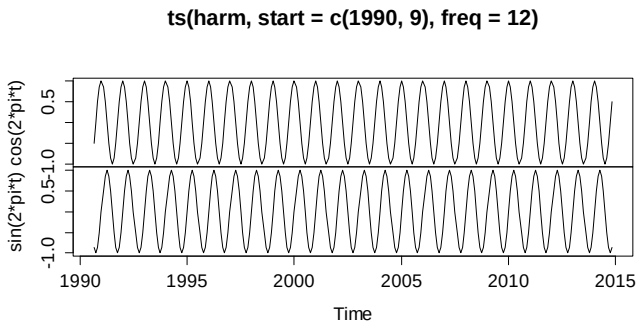
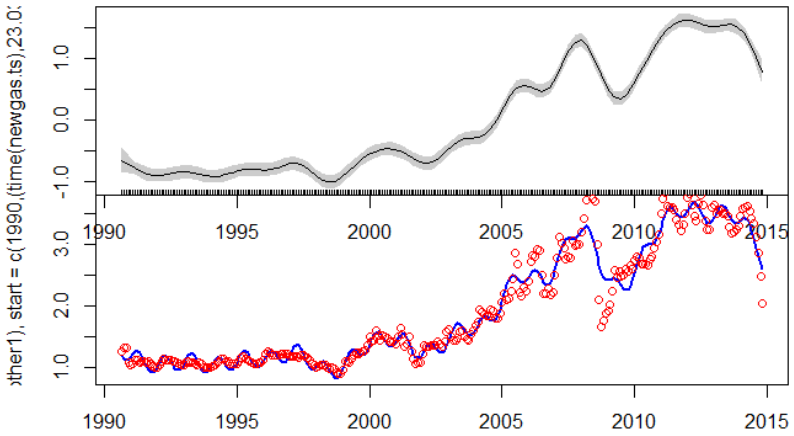


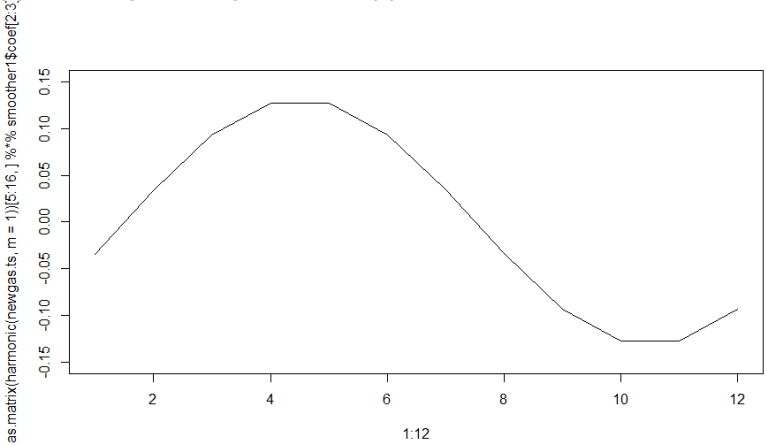

```
> require(mgcv); require(TSA)
> harm<-harmonic(newgas.ts)
> plot(ts(harm,start=c(1990,9),freq=12))
>
> smoother1<-gam(newgas.ts~s(time(newgas.ts),k=25,bs="ts")+harm[,1]+harm[,2])
> summary(smoother1)

Family: gaussian
Link function: identity

Parametric coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.95631    0.01158 168.992  < 2e-16 ***
harm[, 1]   -0.03459    0.01647  -2.101   0.0366 *
harm[, 2]    0.12720    0.01642   7.745 2.04e-13 ***

Approximate significance of smooth terms:
                     edf Ref.df     F p-value
s(time(newgas.ts)) 23.03     24 258.2  <2e-16 ***

R-sq.(adj) =  0.956   Deviance explained =   96%
GCV = 0.042821  Scale est. = 0.03899   n = 291
```

```
> anova(smoother1)
Parametric Terms:
          df      F  p-value
harm[, 1]  1  4.413   0.0366
harm[, 2]  1 59.978 2.04e-13

Approximate significance of smooth terms:
                     edf Ref.df     F p-value
s(time(newgas.ts)) 23.03  24.00 258.2  <2e-16
> smoother1<-gam(newgas.ts~s(time(newgas.ts),k=25,bs="ts")+harmonic(newgas.ts))
> anova(smoother1)

Parametric Terms:
                    df    F  p-value
harmonic(newgas.ts)  2 32.2 3.04e-13

Approximate significance of smooth terms:
                     edf Ref.df     F p-value
s(time(newgas.ts)) 23.03  24.00 258.2  <2e-16
> plot(smoother1,shade=T)
> plot(ts(fitted(smoother1),start=c(1990,9),freq=12),lwd=2,col="blue")
> points(newgas.ts~time(newgas.ts),col="red")
```



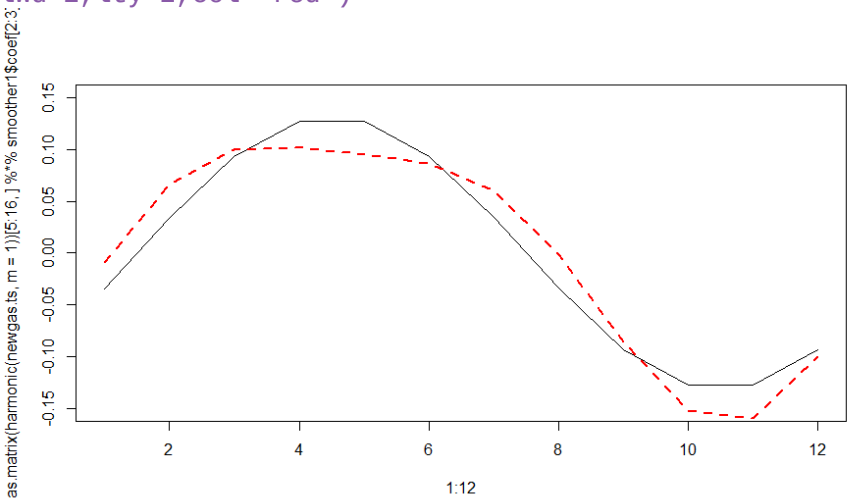


```
> smoother1$coef
                   (Intercept) harmonic(newgas.ts)cos(2*pi*t)
                    1.95630762                    -0.03459329
harmonic(newgas.ts)sin(2*pi*t)           s(time(newgas.ts)).1
                    0.12719974                    -2.90360835
           s(time(newgas.ts)).2           s(time(newgas.ts)).3
                    4.91837438                    -1.02252908
           s(time(newgas.ts)).4           s(time(newgas.ts)).5
                   -2.40372295                    -1.18928628
           s(time(newgas.ts)).6           s(time(newgas.ts)).7
                   -3.25907998                    -2.02018084
           s(time(newgas.ts)).8           s(time(newgas.ts)).9
                    2.52076903                     0.78917412
          s(time(newgas.ts)).10          s(time(newgas.ts)).11
                   -2.21158942                     0.02243476
          s(time(newgas.ts)).12          s(time(newgas.ts)).13
                   -1.13596581                    -0.95874122
          s(time(newgas.ts)).14          s(time(newgas.ts)).15
                    2.47909124                     1.50829346
          s(time(newgas.ts)).16          s(time(newgas.ts)).17
                   -3.50036900                    -1.48279676
          s(time(newgas.ts)).18          s(time(newgas.ts)).19
                    1.50034127                    -0.27615612
          s(time(newgas.ts)).20          s(time(newgas.ts)).21
                   -1.89295330                     1.40703850
          s(time(newgas.ts)).22          s(time(newgas.ts)).23
                   -3.00119715                     7.41981098
          s(time(newgas.ts)).24
                   -4.50152598
> harm[1:12,1:2]%*%smoother1$coef[2:3]
             [,1]
 [1,] -0.09286156
 [2,] -0.12719974
 [3,] -0.12745485
 [4,] -0.09355854
 [5,] -0.03459329
 [6,]  0.03364120
 [7,]  0.09286156
 [8,]  0.12719974
 [9,]  0.12745485
[10,]  0.09355854
[11,]  0.03459329
[12,] -0.03364120
> plot(1:12,as.matrix(harmonic(newgas.ts,m=1))[5:16,]%*%smoother1$coef[2:3],t
ype="l",ylim=c(-.15,.15))
```

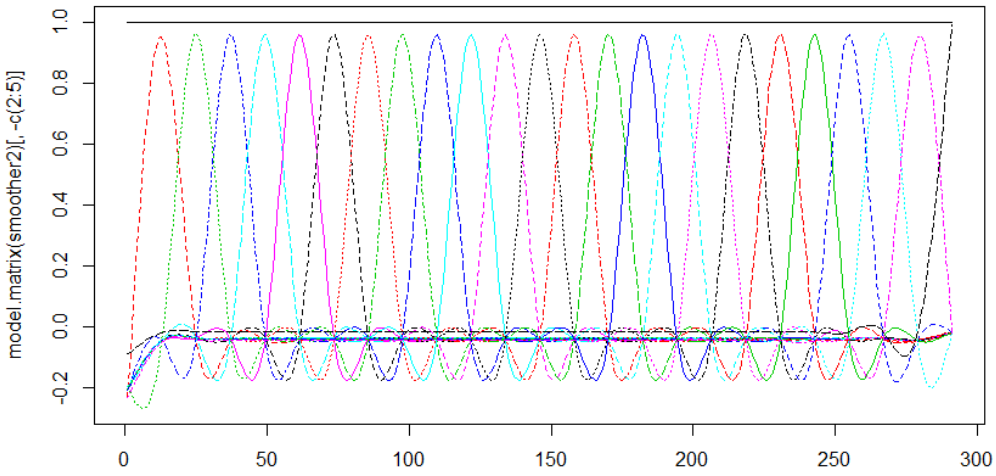


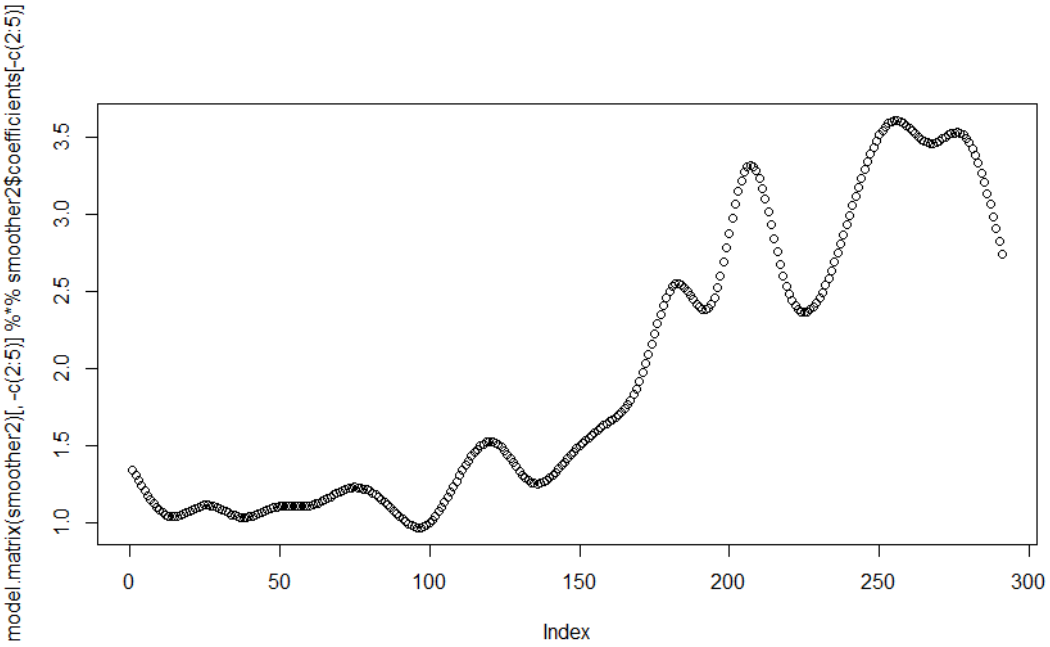

```
> smoother2<-gam(newgas.ts~s(time(newgas.ts),k=25,bs="ts")+harmonic(newgas.ts,m=2))
> summary(smoother2)
```

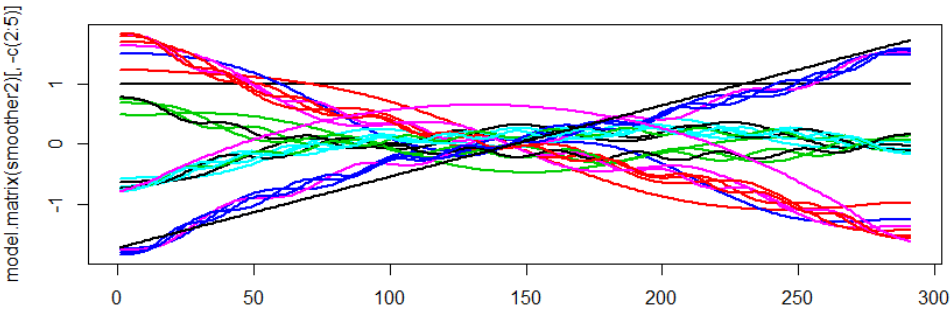
```
                                    Estimate Std. Error t value Pr(>|t|)
(Intercept)                          1.95648    0.01153 169.708  < 2e-16 **
harmonic(newgas.ts, m = 2)cos(2*pi*t) -0.03445    0.01640  -2.101   0.0366 *
harmonic(newgas.ts, m = 2)cos(4*pi*t)  0.02529    0.01634   1.548   0.1228
harmonic(newgas.ts, m = 2)sin(2*pi*t)  0.12707    0.01636   7.769 1.78e-13 **
harmonic(newgas.ts, m = 2)sin(4*pi*t)  0.02225    0.01630   1.365   0.1736

Approximate significance of smooth terms:
                     edf Ref.df     F p-value
s(time(newgas.ts)) 23.04     24 260.4  <2e-16 ***

R-sq.(adj) =  0.956   Deviance explained =   96%
GCV = 0.042788  Scale est. = 0.038665  n = 291
> plot(1:12,as.matrix(harmonic(newgas.ts,m=1))[5:16,]%*%smoother1$coef[2:3],t
ype="l",ylim=c(-.15,.15))
> lines(1:12,as.matrix(harmonic(newgas.ts,m=2))[5:16,]%*%smoother2$coef[2:5],
lwd=2,lty=2,col="red")
```

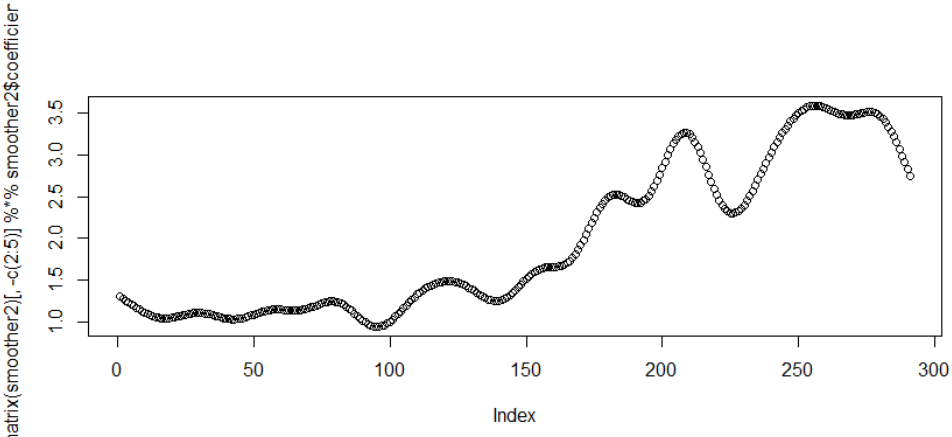



```
> matplot(model.matrix(smoother2)[,-c(2:5)],type="l")
> plot(model.matrix(smoother2)[,-c(2:5)]%*%smoother2$coefficients[-c(2:5)])
```










```
> smoother2<-gam(newgas.ts~s(time(newgas.ts),k=25,bs="ts")+harmonic(newgas.ts
,m=2))
> par(mfrow=c(2,1))
> matplot(model.matrix(smoother2)[,-c(2:5)],type="l",lty=1,lwd=2)
> plot(model.matrix(smoother2)[,-c(2:5)]%*%smoother2$coefficients[-c(2:5)])
```

Another seasonal component modeling technique:

Circular (Penalized) Regression Splines (Wood, 2006; mgcv package in R):

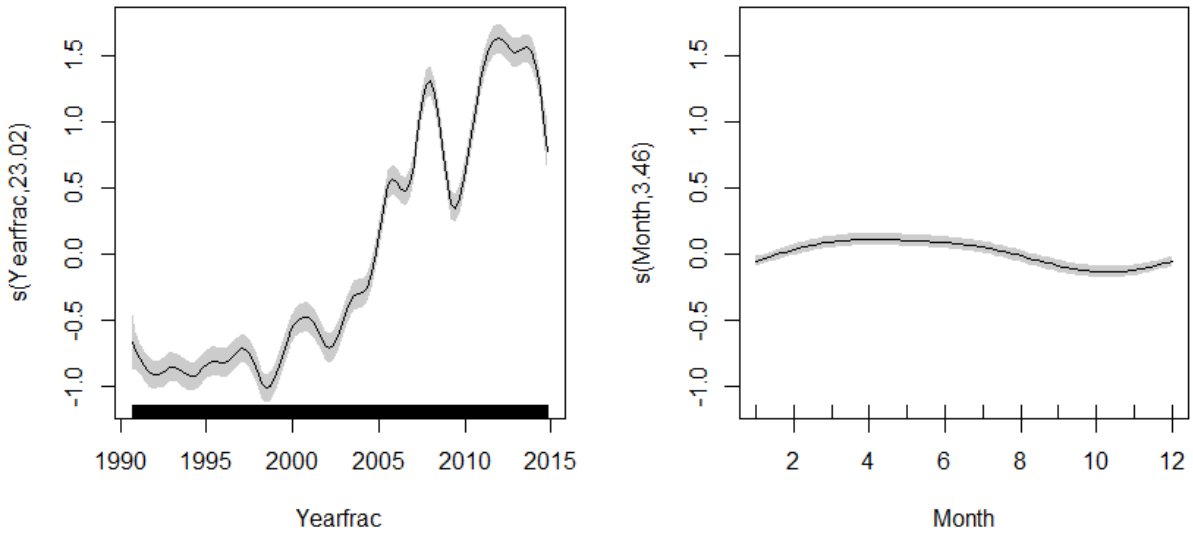
- The main feature of the sine/cosine method is that it is defined to match the estimate at the end of one season and the beginning of the next one.

- Using a suite of sine/cosine bases provides a relatively flexible estimate with the slope coefficients determine the final shape of the curve.
    - Some are relatively unnecessary and so can waste degrees of freedom in the model
- It is possible to define a set of spline basis functions as a function of "x" that have matching estimates at the min and max of the observed x's.
    - The "trick" is to define a basis function that spans the min and max observations to make the functions smooth across this gap (shown above).

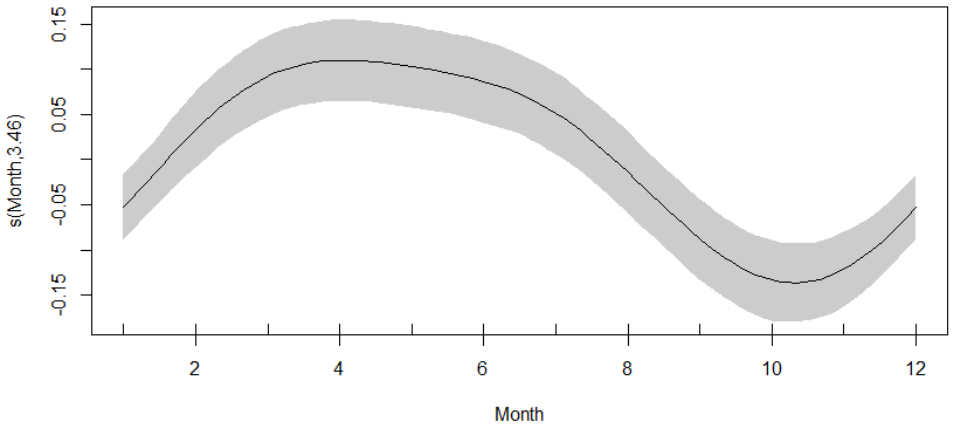
- Otherwise, estimation is like any other penalized regression spline – with the constraint now of matching min and max estimates.
- Smoothing of the effect can lead to more efficient estimates of smooth but not sinusoidal seasonal effects than either the sine/cosine decomposition or seasonal means.
- Picking the initial number of basis functions (k=? controls this in s(.), with 10 the default choice) constrains the potential roughness of the function.

    R notation:
    `s(cycle(tsobject),bs="cc",k=length(unique(cycle(tsobject)))`

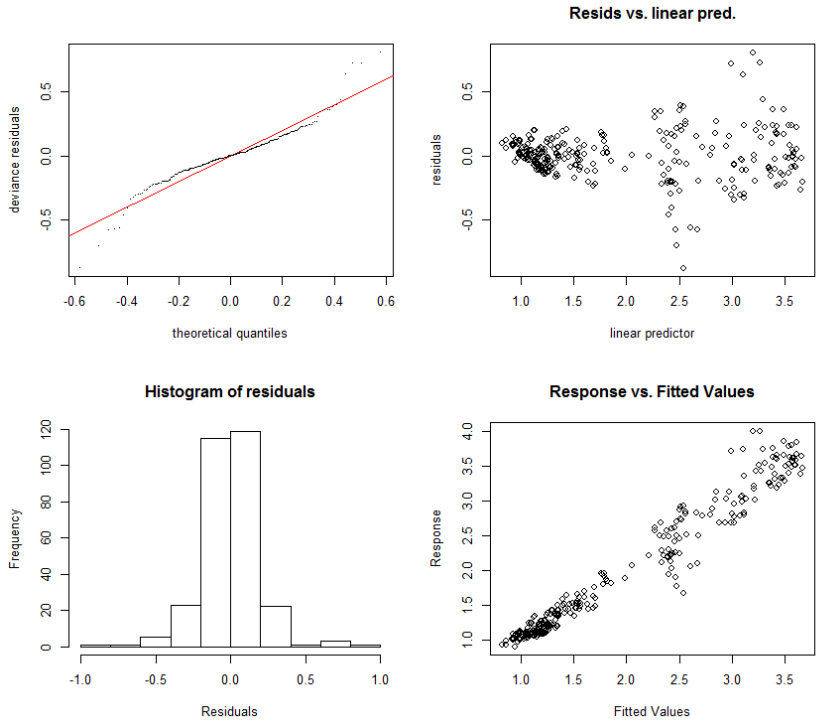
```
> data1<-data.frame(Y=as.vector(newgas.ts),Yearfrac=time(newgas.ts),Month=cyc
le(newgas.ts))
> smoother3<-gam(Y~s(Yearfrac,k=25,bs="ts")+s(Month,bs="cc",k=12),data=data1)
> summary(smoother3)
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   1.9551     0.0116   168.6   <2e-16 ***
               edf Ref.df       F  p-value
s(Yearfrac) 23.017     24 257.335  < 2e-16 ***
s(Month)     3.463     10   5.966 1.42e-13 ***

R-sq.(adj) =  0.956   Deviance explained =   96%
GCV = 0.043208  Scale est. = 0.039127  n = 291
```



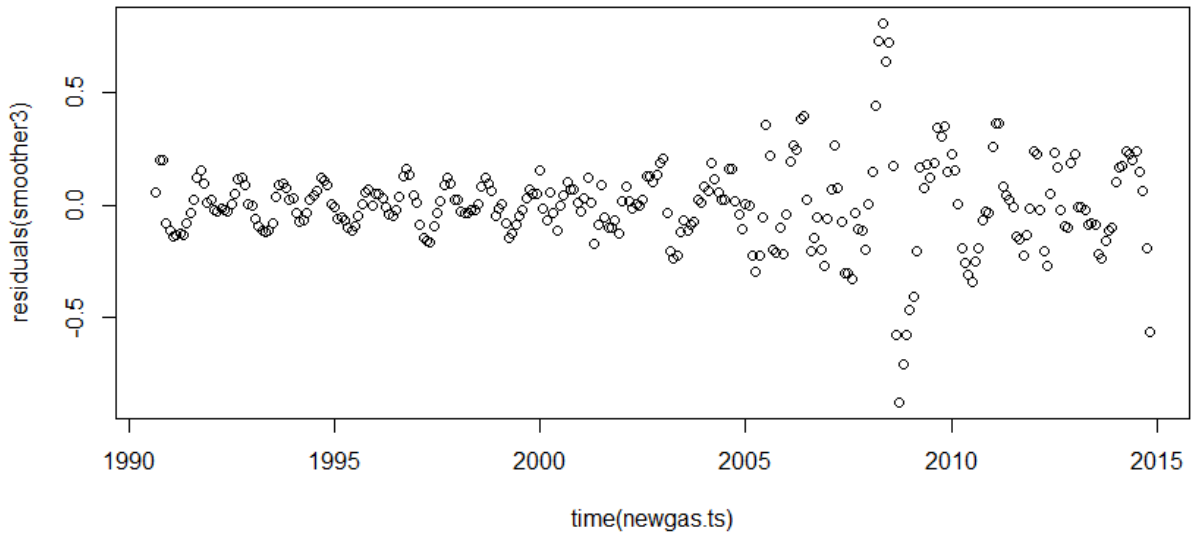

```
> plot(smoother3,scale=0,shade=T)
```



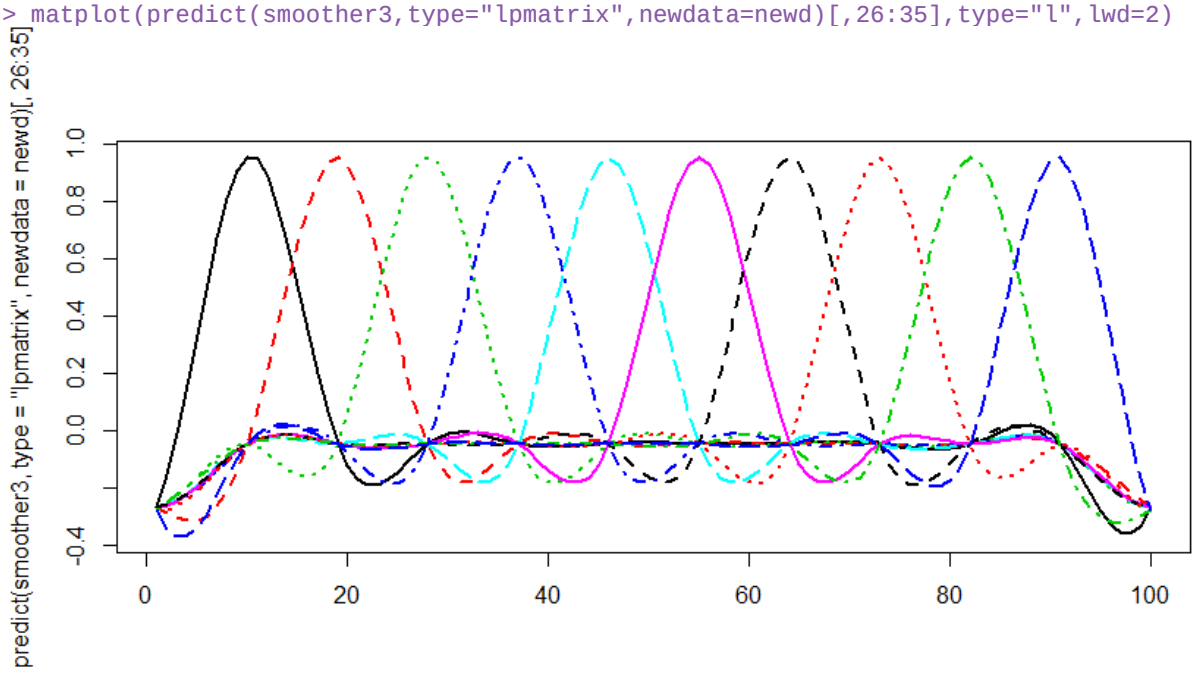

```
> gam.check(smoother3)

Method: GCV   Optimizer: magic
Smoothing parameter selection converged after 10 iterations.
The RMS GCV score gradiant at convergence was 2.477261e-06 .
The Hessian was positive definite.
The estimated model rank was 35 (maximum possible: 35)
Model rank =  35 / 35

Basis dimension (k) checking results. Low p-value (k-index<1) may
indicate that k is too low, especially if edf is close to k'.

                k'    edf k-index p-value
s(Yearfrac) 24.000 23.017   0.223    0.00
s(Month)    10.000  3.463   1.073    0.88
```




```
> plot(residuals(smoother3)~time(newgas.ts))
```




```
> newd<-data.frame(Yearfrac=rep(2015,100),Month=seq(from=1,to=12,length.out=100))
> matplot(predict(smoother3,type="lpmatrix",newdata=newd)[,26:35],type="l",lwd=2)
```



CC 3.4: Reliability and Efficiency of Regression Estimates:

- In this section, CC do a lot of work to derive the variance of different estimators such as a seasonal mean or a cosine trend for a particular spot in the cycle or the slope coefficient
  - While interesting, I think this section misses the mark a little.

- For large samples, the typical least squares estimators often have approximately the same variance as the best linear unbiased estimators (BLUE).
  - consider $n \to \infty$ in our constant mean examples?

- But it is much easier (especially for reviewers) to have attempted to identify and then accommodate the autocorrelation structure instead of arguing that your estimator was unbiased and had close to the correct variance in a finite sample situation