

ExplorationPhasor

J.Godet

13/10/2020

effect of wavelength position

```
require(spectralPhasor)

## Loading required package: spectralPhasor

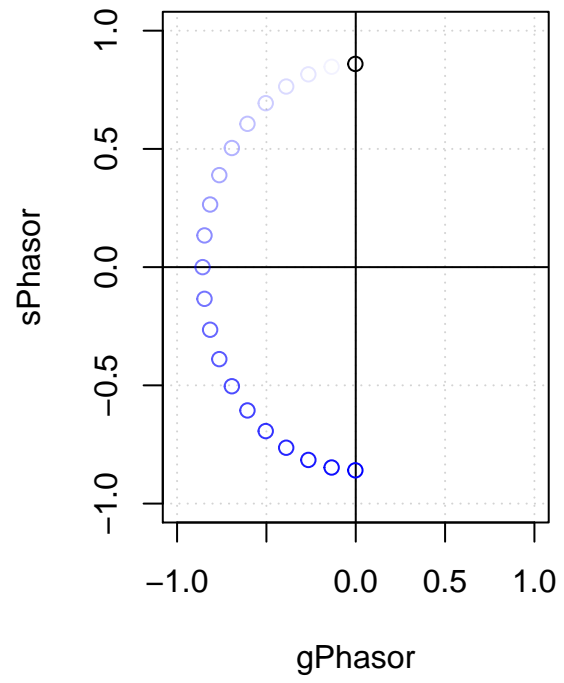
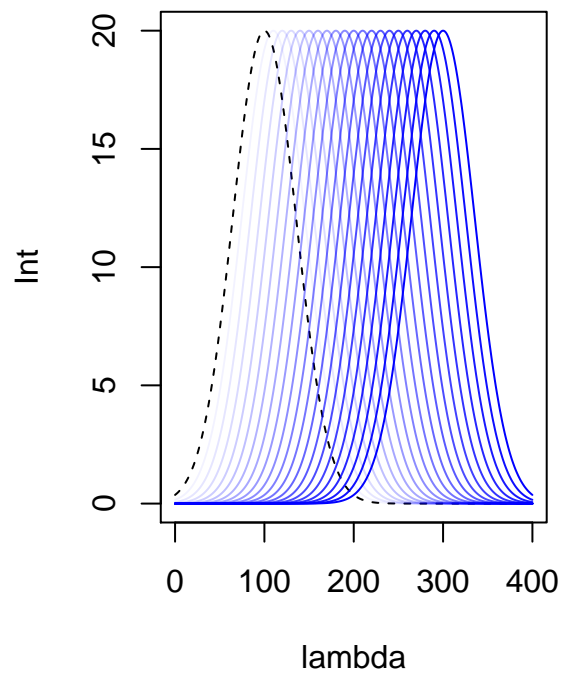
Int <- 20* exp(-(0:400 - 100)^2/50^2)
lambda <- seq(0,400,length=401)

par(mfrow=c(1,2))
plot(lambda, Int, type='l', lty=2)
for (i in 1:20){
  Int <- 20* exp(-(0:400 - (100+10*i))^2/50^2)
  lines(lambda, Int, col=rgb(0,0,1, i/20))
}

Int <- 20* exp(-(0:400 - 100)^2/50^2)
lambda <- seq(0,400,length=401)

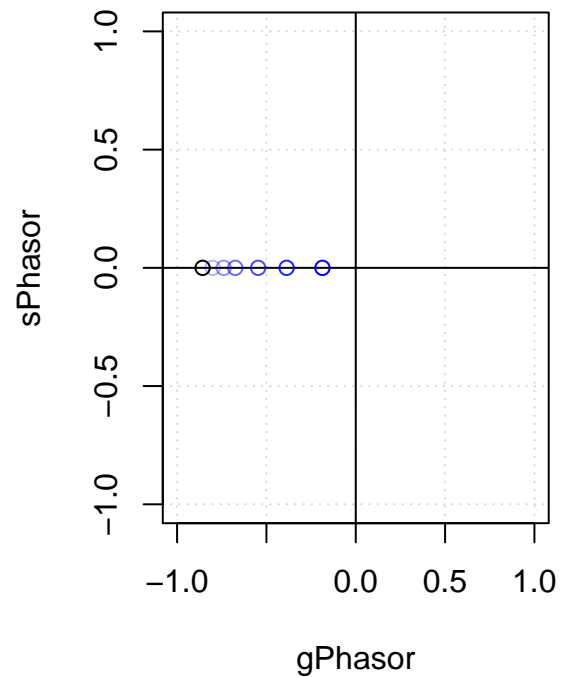
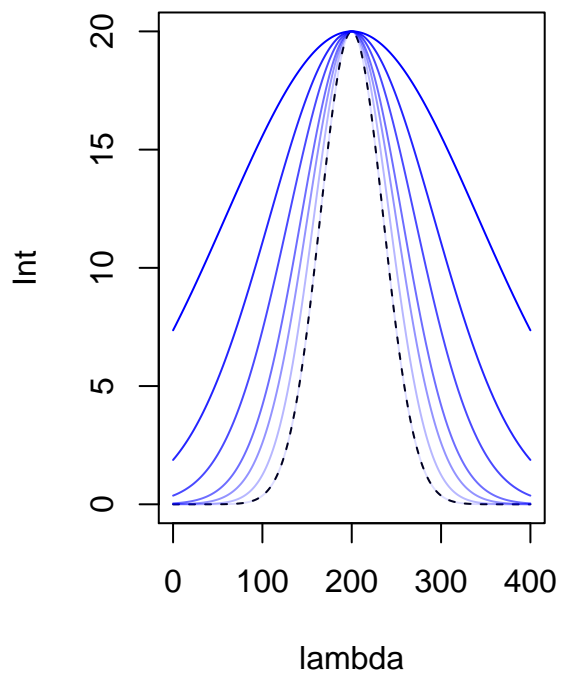
g = calcPhasor(Int = Int,lambda = lambda)[1]
s = calcPhasor(Int = Int,lambda = lambda)[2]

plotPhasor(g,s, xlim=c(-1,1), ylim=c(-1,1))
for (i in 1:20){
  Int <- 20* exp(-(0:400 - (100+10*i))^2/50^2)
  points(calcPhasor(Int = Int,lambda = lambda), col=rgb(0,0,1, i/20))
}
```



effect of spectra width

```
par(mfrow=c(1,2))
width <- c(50,60,70,80,100,130,200)
Int <- 20* exp(-(0:400 - 200)^2/50^2)
lambda <- seq(0,400,length=401)
plot(lambda, Int, type='l', lty=2)
for (i in 1:7){
  Int <- 20* exp(-(0:400 - 200)^2/width[i]^2)
  lines(lambda, Int, col=rgb(0,0,1, i/7))
}
for (i in 1:7){
  Int <- 20* exp(-(0:400 - 200)^2/width[i]^2)
  if(i<2){
    g = calcPhasor(Int = Int,lambda = lambda)[1]
    s = calcPhasor(Int = Int,lambda = lambda)[2]
    plotPhasor(g,s, xlim=c(-1,1), ylim=c(-1,1))
  }else{
    points(calcPhasor(Int = Int,lambda = lambda), col=rgb(0,0,1, i/7))
  }
}
```



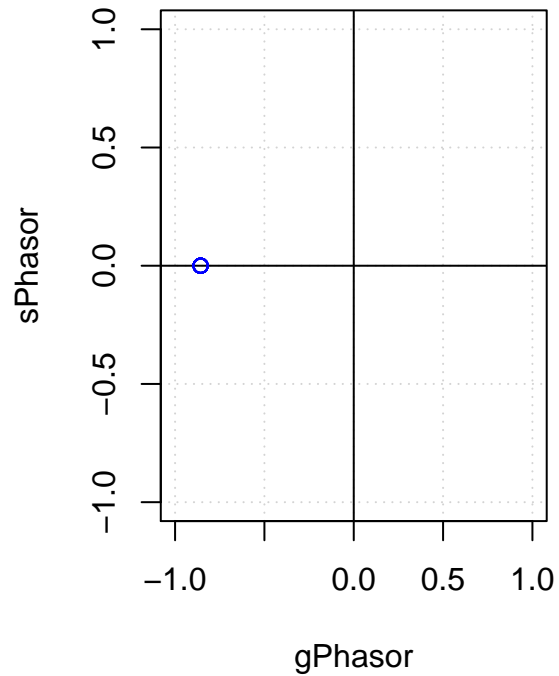
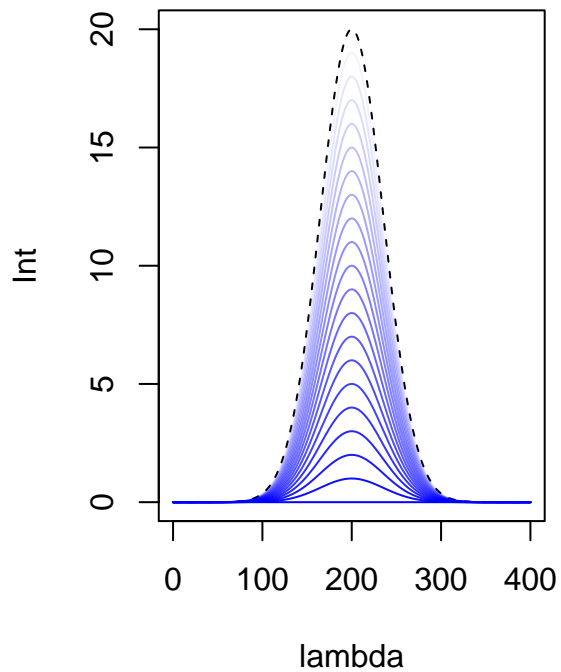
effect of spectra intensity

```
Int <- 20* exp(-(0:400 - 200)^2/50^2)
lambda <- seq(0,400,length=401)

par(mfrow=c(1,2))
plot(lambda, Int, type='l', lty=2)
for (i in 1:20){
  Int0 <- Int * (100-5*i)/100
  lines(lambda, Int0, col=rgb(0,0,1, i/20))
}

g = calcPhasor(Int = Int,lambda = lambda)[1]
s = calcPhasor(Int = Int,lambda = lambda)[2]

plotPhasor(g,s, xlim=c(-1,1), ylim=c(-1,1))
for (i in 1:20){
  Int0 <- Int * (100-5*i)/100
  points(calcPhasor(Int = Int0,lambda = lambda), col=rgb(0,0,1, i/20))
}
```

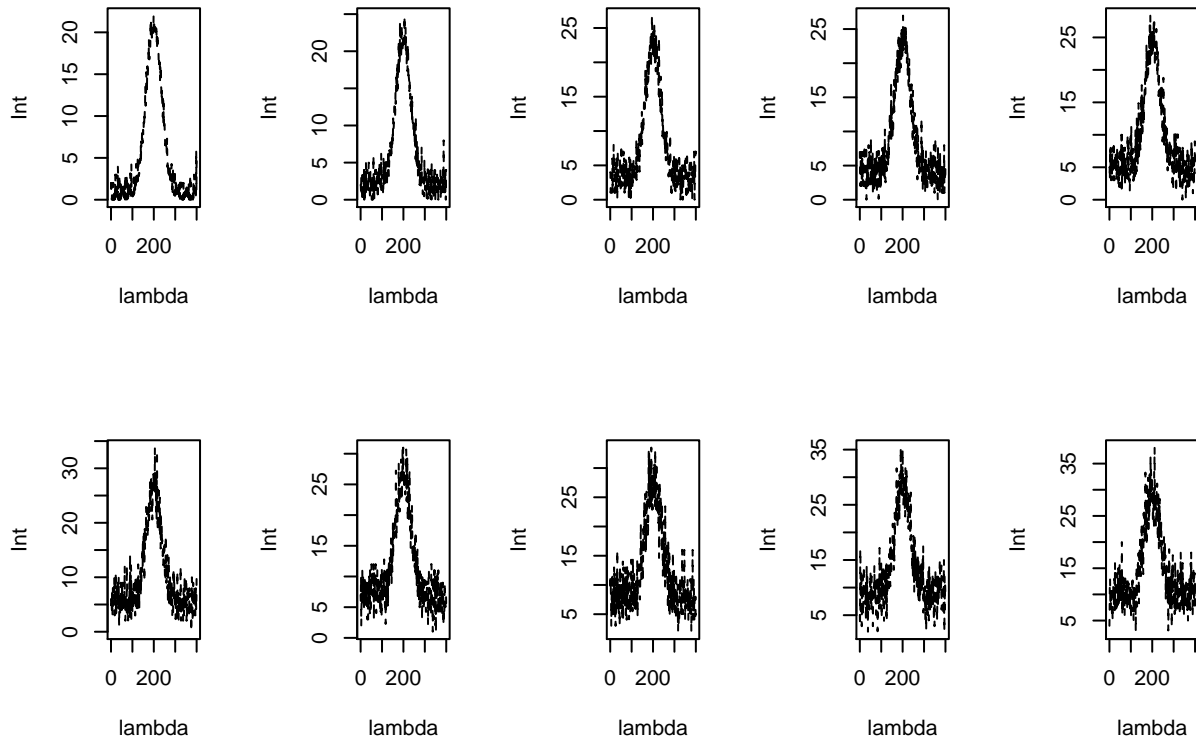


effect of noise on spectra intensity

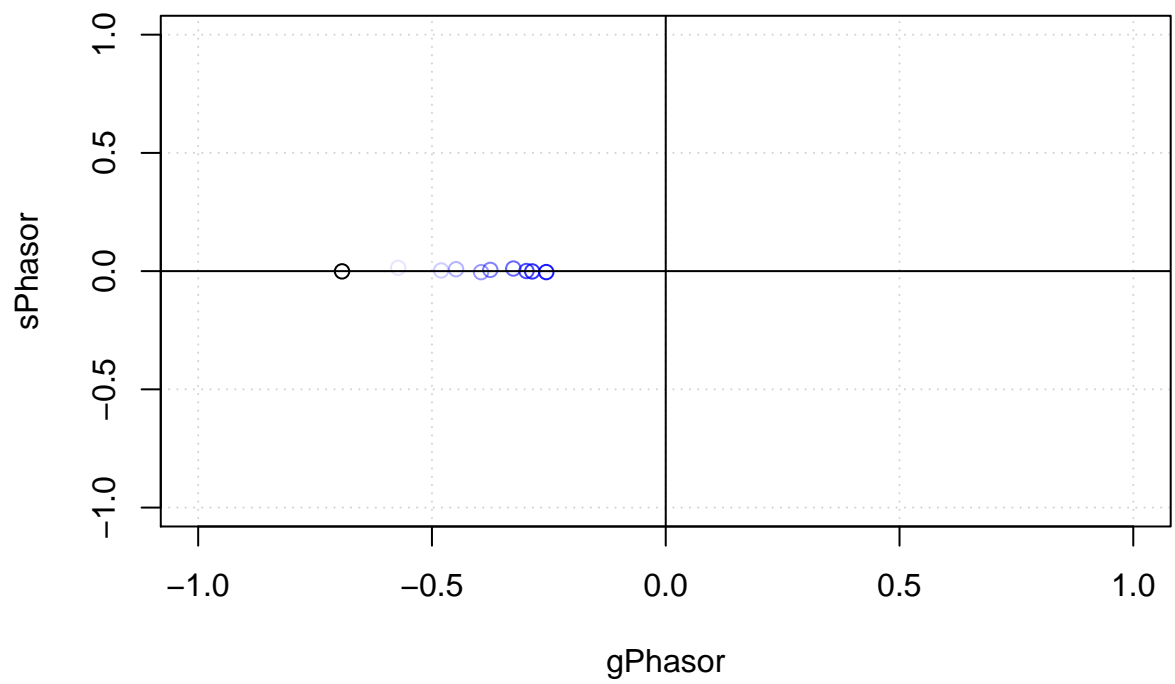
```
par(mfrow=c(2,5))
g <- s <- numeric()

for (i in 0:10){

  Int <- 20* exp(-(0:400 - 200)^2/50^2) + rpois(n = 401, lambda = i)
  lambda <- seq(0,400,length=401)
  if(i>0){plot(lambda, Int, type='l', lty=2)}
  g[i] = calcPhasor(Int = Int,lambda = lambda)[1]
  s[i] = calcPhasor(Int = Int,lambda = lambda)[2]
}
```



```
par(mfrow=c(1,1))
plotPhasor(g,s, xlim=c(-1,1), ylim=c(-1,1), col=c("black",rgb(0,0,1,1:10/10)))
```



effect of profile length

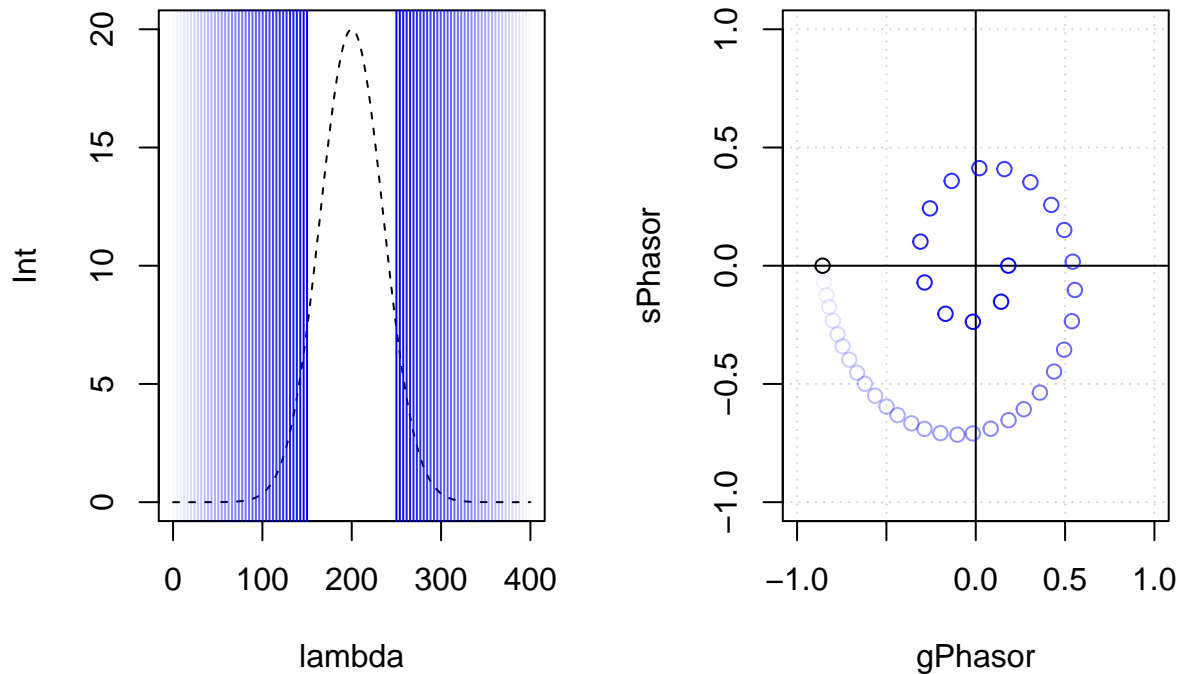
```
Int <- 20* exp(-(0:400 - 200)^2/50^2)
lambda <- seq(0,400,length=401)

par(mfrow=c(1,2))
plot(lambda, Int, type='l', lty=2)

cutPix <- seq(1,150,length=40)
for (i in 1:40){
  abline(v = cutPix[i], col=rgb(0,0,1,i/40))
  abline(v = 400-cutPix[i], col=rgb(0,0,1,i/40))
}

g = calcPhasor(Int = Int,lambda = lambda)[1]
s = calcPhasor(Int = Int,lambda = lambda)[2]

plotPhasor(g,s, xlim=c(-1,1), ylim=c(-1,1))
for (i in 1:40){
  points(calcPhasor(Int = Int[cutPix[i):(400-cutPix[i])],lambda = lambda[cutPix[i):(400-cutPix[i])]]), c
}
```



Asymmetric position

```
Int <- 20* exp(-(0:400 - 150)^2/50^2)
lambda <- seq(0,400,length=401)

par(mfrow=c(1,2))
plot(lambda, Int, type='l', lty=2)

cutPix <- seq(1,150,length=40)
for (i in 1:40){
```

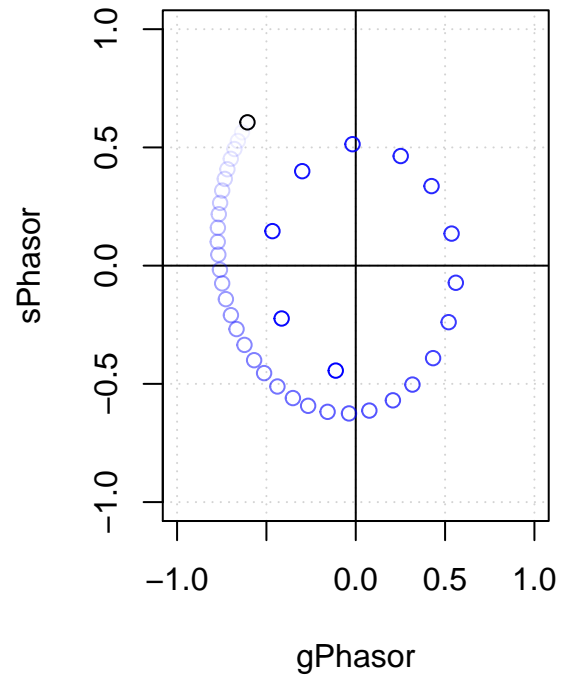
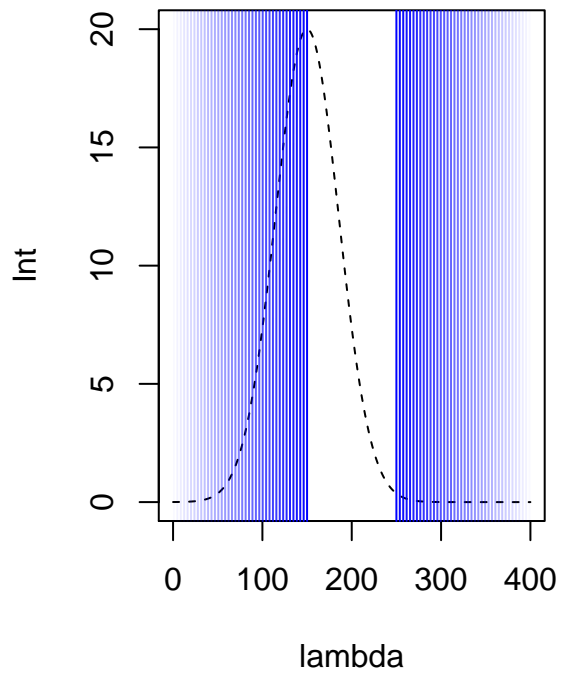
```

abline(v = cutPix[i], col=rgb(0,0,1,i/40))
abline(v = 400-cutPix[i], col=rgb(0,0,1,i/40))
}

g = calcPhasor(Int = Int,lambda = lambda)[1]
s = calcPhasor(Int = Int,lambda = lambda)[2]

plotPhasor(g,s, xlim=c(-1,1), ylim=c(-1,1))
for (i in 1:40){
  points(calcPhasor(Int = Int[cutPix[i):(400-cutPix[i])],lambda = lambda[cutPix[i):(400-cutPix[i])]), c
}

```

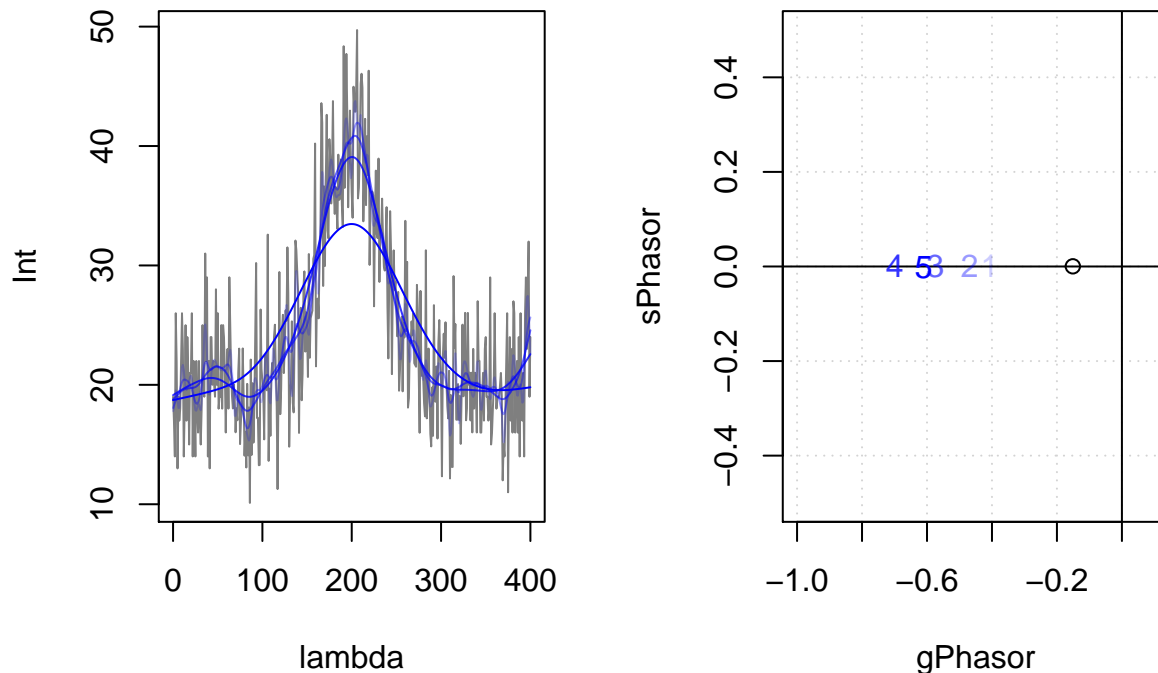


effect of denoising spectra

```
par(mfrow=c(1,2))
g <- s <- numeric()

Int <- 20* exp(-(0:400 - 200)^2/50^2) + rpois(n = 401, lambda = 20)
lambda <- seq(0,400,length=401)
plot(lambda, Int, type='l', col=rgb(0,0,0,.5))
dfList <- c(100,50,20,10,5)
for (i in 1:5){
  aa <- smooth.spline(x = lambda, y = Int, df=dfList[i])
  lines(aa,col=rgb(0,0,1, i/5))
}

g = calcPhasor(Int = Int ,lambda = lambda)[1]
s = calcPhasor(Int = Int ,lambda = lambda)[2]
plotPhasor(g,s, xlim=c(-1,.1), ylim=c(-.5,.5), col=c("black"))
for (i in 1:5){
  aa <- smooth.spline(x = lambda, y = Int, df=dfList[i])
  g = calcPhasor(Int = aa$y-min(aa$y),lambda = lambda)[1]
  s = calcPhasor(Int = aa$y-min(aa$y),lambda = lambda)[2]
  print(calcAngle(g,s))
  points(g,s, col=rgb(0,0,1, i/5), pch=paste(i))
}
```



```
## [1] 179.8766
## [1] 179.8768
## [1] 179.8795
## [1] 179.9083
## [1] 180.1289
```

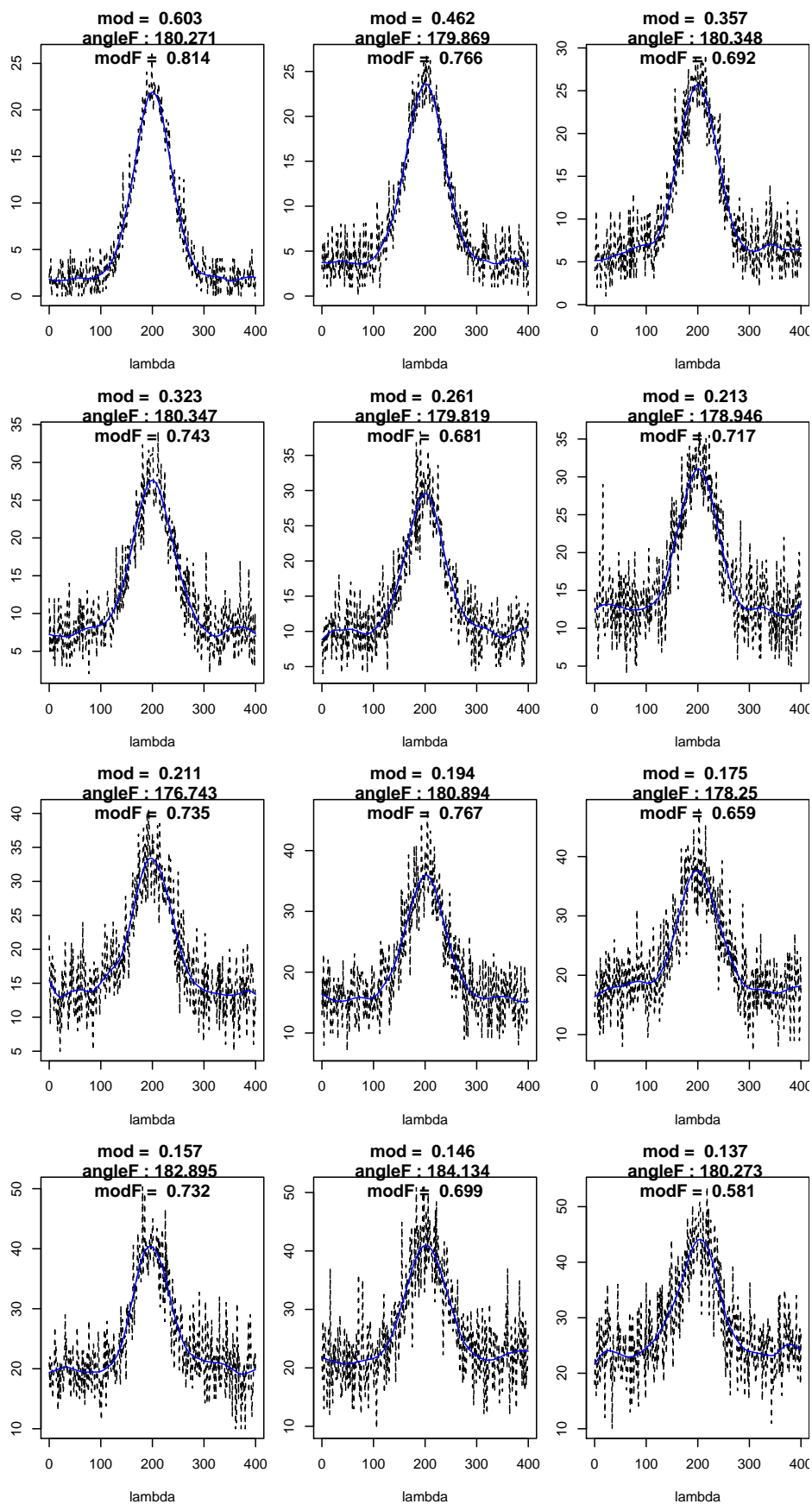
No adjust on df - use automatic df from the spline function

```

par(mfrow=c(4,3), mar=c(5,3,2,0))
g <- s <- numeric()

for (i in 1:12){
  Int <- 20* exp(-(0:400 - 200)^2/50^2) + rpois(n = 401, lambda = 2*i)
  lambda <- seq(0,400,length=401)
  plot(lambda, Int, type='l', lty=2)
  aa <- smooth.spline(x = lambda, y = Int)
  lines(aa,col=rgb(0,0,1))
  g = calcPhasor(Int = Int,lambda = lambda)[1]
  s = calcPhasor(Int = Int,lambda = lambda)[2]
  gaft = calcPhasor(Int = aa$y-min(aa$y),lambda = lambda)[1]
  saft = calcPhasor(Int = aa$y-min(aa$y),lambda = lambda)[2]
  title(cex=.5, main=paste("angle :",round(calcAngle(g, s),3), "\nmod = ", round(calcModule(g,s),3),
    "\nangleF :",round(calcAngle(gaft, saft),3), "\nmodF = ", round(calcModule(gaft,saft),3),
    ))
}

```



No adjust on df - use automatic df from the spline function on lower and lower snr

```
par(mfrow=c(4,3), mar=c(5,3,2,0))
g <- s <- numeric()

for (i in 1:12){
  Int <- (20*(1-i/13)* exp(-(0:400 - 200)^2/50^2) + rpois(n = 401, lambda = 20))[100:300]
  lambda <- seq(0,400,length=401)[100:300]
  plot(lambda, Int, type='l', lty=2)
  aa <- smooth.spline(x = lambda, y = Int)
  lines(aa,col=rgb(0,0,1))
  g = calcPhasor(Int = Int,lambda = lambda)[1]
  s = calcPhasor(Int = Int,lambda = lambda)[2]
  gaft = calcPhasor(Int = aa$y-min(aa$y),lambda = lambda)[1]
  saft = calcPhasor(Int = aa$y-min(aa$y),lambda = lambda)[2]
  title(cex=.5, main=paste("angle :",round(calcAngle(g, s),3), "mod = ", round(calcModule(g,s),3),
    "\nangleF :",round(calcAngle(gaft, saft),3), "modF = ", round(calcModule(gaft,saft),3),
    ))
}
```

