Travail sur les SEM

myGroup <- "Social"  
  
## EFA with polychoric FA (iterative process: change the number of factors and observe results: do they make sense?)  
myDataFactor <- myDataFacImp[ , allSelections[[myGroup]]]  
myCor <- hetcor(myDataFactor) # polychoric corr matrix

Warning in FUN(X[[i]], ...): polychoric correlation between variables V116 and V118 produced warnings:  
 NaNs produced  
 NaNs produced

I checked out this error and couldn’t find a reason for it.

For myGroup= “Social”, we get : “polychoric correlation between variables V116 and V118 produced warnings: NaNs produced.”

I looked at these 2 vars more specifically. The NaN warning persists, but results seem consistent and I can’t locate the NaNs.

dat <- myDataFactor[,c("V116", "V118")]  
str(dat)

'data.frame': 82 obs. of 2 variables:  
 $ V116: Ord.factor w/ 3 levels "V116\_0"<"V116\_1"<..: 2 2 2 3 2 3 3 3 2 1 ...  
 $ V118: Ord.factor w/ 5 levels "V118\_0"<"V118\_1"<..: 3 3 3 4 2 4 4 4 2 2 ...

hetcor(dat)

Warning in FUN(X[[i]], ...): polychoric correlation between variables V116 and V118 produced warnings:  
 NaNs produced  
 NaNs produced

Two-Step Estimates  
  
Correlations/Type of Correlation:  
 V116 V118  
V116 1 Polychoric  
V118 0.8603 1  
  
Standard Errors:  
[1] "" "0.04495"  
  
n = 82   
  
P-values for Tests of Bivariate Normality:  
[1] "" "0.05806"

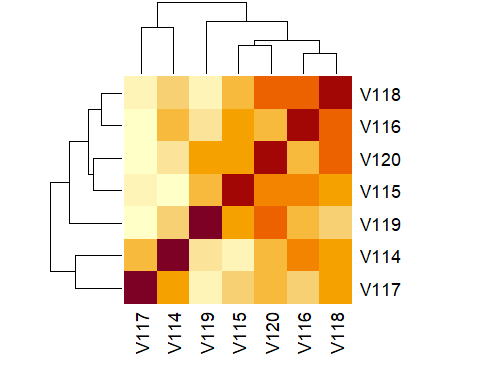
# But the 2 vars are highly correlated...   
  
# Tried to use another function:  
polychor(dat$V116, dat$V118) # no errors here. Same high corr coefficient. So that might be the pb.

[1] 0.8603013

Check if the ML method would be better:

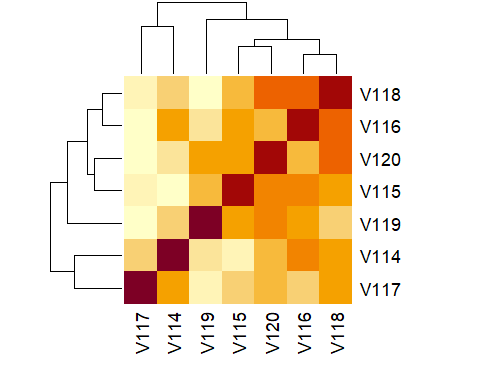
myCor2 <- hetcor(myDataFactor, ML=TRUE) # no errors here.

# Compare outputs: they're very similar. Here’s the original corr matrix:  
heatmap(myCor$correlations)



they're very similr, And the one calculated by ML: :

heatmap(myCor2$correlations)



CCL: I wouldn’t worry too much. You can use ML if you’d prefer, but it’s much slower, and since the results are pretty much the same…

# Next step: EFA.

n=82  
# this produces some warnings about ultra-heywood cases.  
myEFA <- psych::fa(r=myCor$correlations, nfactors = 2, n.obs=n, rotate = "varimax", fm="ml")

from here: <https://www.sfu.ca/sasdoc/sashtml/stat/chap26/sect21.htm>

ultra-Heywood case implies that some unique factor has negative variance, a clear indication that something is wrong. Possible causes include

* bad prior communality estimates
* too many common factors
* too few common factors
* not enough data to provide stable estimates
* the common factor model is not an appropriate model for the data

I can’t be sure about the 1st point. It could be related to the EFA warning, but I don’t think so. I tried to run the command again after selecting a subset of factors for which there were no “NaNs produced” warnings, and I still got this.

Too many / few common factors : I checked this, and it looks fine:

# to find out how many factors are needed: 2. All good. I tried a few different "fm" methods, without any major improvements...

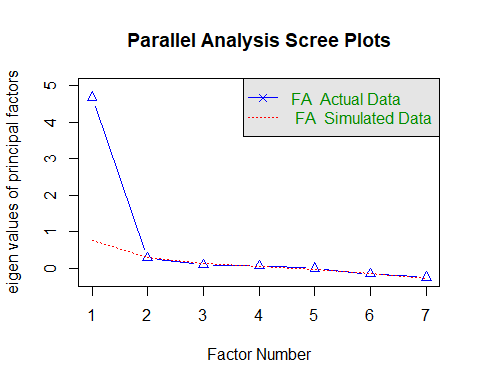
parallel<-fa.parallel(myCor$correlations, fm='minres', fa='fa')

Warning in fa.parallel(myCor$correlations, fm = "minres", fa = "fa"): It seems  
as if you are using a correlation matrix, but have not specified the number of  
cases. The number of subjects is arbitrarily set to be 100

Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :  
The estimated weights for the factor scores are probably incorrect. Try a  
different factor score estimation method.

Warning in fac(r = r, nfactors = nfactors, n.obs = n.obs, rotate = rotate, : An  
ultra-Heywood case was detected. Examine the results carefully

Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :  
The estimated weights for the factor scores are probably incorrect. Try a  
different factor score estimation method.



Parallel analysis suggests that the number of factors = 1 and the number of components = NA

So I think the ultra-Heywood issue it has to do with the 2 last points: not enough data and/or CF model inappropriate.

I also tried other functions:

# use lavCor instead of fa():  
cc\_cor <- lavaan::lavCor(myDataFactor, ordered = unlist(allSelectionsBis[[myGroup]]) )  
  
cc\_cor # roughly the same output as myCor$correlations

V114 V115 V116 V117 V118 V119 V120  
V114 1.000   
V115 0.502 1.000   
V116 0.744 0.785 1.000   
V117 0.625 0.529 0.527 1.000   
V118 0.675 0.750 0.860 0.629 1.000   
V119 0.550 0.685 0.654 0.387 0.595 1.000   
V120 0.630 0.778 0.735 0.545 0.856 0.771 1.000

n=82 # nb of obs  
cc\_EFA <- psych::fa(r=cc\_cor, nfactors = 2, n.obs=n, rotate = "varimax", fm="ml") # no errors or warnings here  
  
# Compare outputs: myEFA vs cc\_EFA. Matches pretty well.  
myEFA

Factor Analysis using method = ml  
Call: psych::fa(r = myCor$correlations, nfactors = 2, n.obs = n, rotate = "varimax",   
 fm = "ml")  
Standardized loadings (pattern matrix) based upon correlation matrix  
 ML1 ML2 h2 u2 com  
V114 0.41 0.63 0.57 0.430 1.7  
V115 0.59 0.60 0.70 0.296 2.0  
V116 0.40 0.91 1.00 0.005 1.4  
V117 0.43 0.39 0.34 0.664 2.0  
V118 0.65 0.66 0.85 0.150 2.0  
V119 0.66 0.43 0.61 0.389 1.7  
V120 0.91 0.40 1.00 0.005 1.4  
  
 ML1 ML2  
SS loadings 2.53 2.53  
Proportion Var 0.36 0.36  
Cumulative Var 0.36 0.72  
Proportion Explained 0.50 0.50  
Cumulative Proportion 0.50 1.00  
  
Mean item complexity = 1.7  
Test of the hypothesis that 2 factors are sufficient.  
  
df null model = 21 with the objective function = 6.84 with Chi Square = 532.43  
df of the model are 8 and the objective function was 0.77   
  
The root mean square of the residuals (RMSR) is 0.06   
The df corrected root mean square of the residuals is 0.1   
  
The harmonic n.obs is 82 with the empirical chi square 13.69 with prob < 0.09   
The total n.obs was 82 with Likelihood Chi Square = 59.14 with prob < 6.9e-10   
  
Tucker Lewis Index of factoring reliability = 0.733  
RMSEA index = 0.279 and the 90 % confidence intervals are 0.216 0.35  
BIC = 23.89  
Fit based upon off diagonal values = 0.99  
Measures of factor score adequacy   
 ML1 ML2  
Correlation of (regression) scores with factors 0.99 0.99  
Multiple R square of scores with factors 0.99 0.99  
Minimum correlation of possible factor scores 0.98 0.98

cc\_EFA

Factor Analysis using method = ml  
Call: psych::fa(r = cc\_cor, nfactors = 2, n.obs = n, rotate = "varimax",   
 fm = "ml")  
Standardized loadings (pattern matrix) based upon correlation matrix  
 ML1 ML2 h2 u2 com  
V114 0.41 0.63 0.57 0.430 1.7  
V115 0.59 0.60 0.70 0.296 2.0  
V116 0.40 0.91 1.00 0.005 1.4  
V117 0.43 0.39 0.34 0.664 2.0  
V118 0.65 0.66 0.85 0.150 2.0  
V119 0.66 0.43 0.61 0.389 1.7  
V120 0.91 0.40 1.00 0.005 1.4  
  
 ML1 ML2  
SS loadings 2.53 2.53  
Proportion Var 0.36 0.36  
Cumulative Var 0.36 0.72  
Proportion Explained 0.50 0.50  
Cumulative Proportion 0.50 1.00  
  
Mean item complexity = 1.7  
Test of the hypothesis that 2 factors are sufficient.  
  
df null model = 21 with the objective function = 6.84 with Chi Square = 532.42  
df of the model are 8 and the objective function was 0.77   
  
The root mean square of the residuals (RMSR) is 0.06   
The df corrected root mean square of the residuals is 0.1   
  
The harmonic n.obs is 82 with the empirical chi square 13.69 with prob < 0.09   
The total n.obs was 82 with Likelihood Chi Square = 59.14 with prob < 6.9e-10   
  
Tucker Lewis Index of factoring reliability = 0.733  
RMSEA index = 0.279 and the 90 % confidence intervals are 0.216 0.35  
BIC = 23.88  
Fit based upon off diagonal values = 0.99  
Measures of factor score adequacy   
 ML1 ML2  
Correlation of (regression) scores with factors 0.99 0.99  
Multiple R square of scores with factors 0.99 0.99  
Minimum correlation of possible factor scores 0.98 0.98

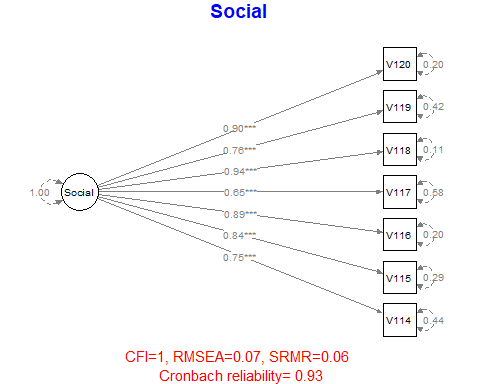
# CFA

For this first bit of code, I replaced the bits producing warnings by the alternative functions from lavaan I mentionned before, which produced the same results, but without the warnings about NaNs.

myDataFactor <- myDataFacImp[ , unlist(allSelectionsBis[[myGroup]])]  
SEM1 <- createModel(myGroup)  
fit1 <- cfa(SEM1, data = myDataFacImp, sample.nobs = n, std.lv = TRUE, ordered = TRUE)  
summary(fit1, standardized = TRUE) # we want the pvalue to be > 0.05 because the model has to be similar to expectations, rather than different

lavaan 0.6-19 ended normally after 16 iterations  
  
 Estimator DWLS  
 Optimization method NLMINB  
 Number of model parameters 31  
  
 Number of observations 82  
  
Model Test User Model:  
 Standard Scaled  
 Test Statistic 20.198 34.543  
 Degrees of freedom 14 14  
 P-value (Chi-square) 0.124 0.002  
 Scaling correction factor 0.652  
 Shift parameter 3.573  
 simple second-order correction   
  
Parameter Estimates:  
  
 Parameterization Delta  
 Standard errors Robust.sem  
 Information Expected  
 Information saturated (h1) model Unstructured  
  
Latent Variables:  
 Estimate Std.Err z-value P(>|z|) Std.lv Std.all  
 Social =~   
 V114 0.749 0.044 17.163 0.000 0.749 0.749  
 V115 0.841 0.034 24.503 0.000 0.841 0.841  
 V116 0.892 0.037 24.101 0.000 0.892 0.892  
 V117 0.648 0.061 10.667 0.000 0.648 0.648  
 V118 0.942 0.040 23.288 0.000 0.942 0.942  
 V119 0.759 0.061 12.419 0.000 0.759 0.759  
 V120 0.895 0.024 37.673 0.000 0.895 0.895

myFitMeasures <- fitMeasures(fit1, c("cfi", "rmsea", "srmr"))  
myFit <- paste(paste(c("CFI", "RMSEA", "SRMR"), round(c(myFitMeasures), 2), sep = "="), collapse = ", ")  
if (!is.list(allSelectionsBis[[myGroup]])) {  
 # myCronbach <- psych::alpha(hetcor(myDataFactor)$correlations, n.obs = n, check.keys=TRUE)   
 # CC: alternative function:  
 cc\_Cronbach <- psych::alpha(lavaan::lavCor(myDataFactor))   
  
 myFit <- paste(myFit, "\n", "Cronbach reliability=", round(cc\_Cronbach$total$raw\_alpha, 2))  
}  
  
plotFit1 <- semPaths(fit1, what = "path", whatLabels = "stand", rotation = 2, intercepts = FALSE, thresholds = FALSE,  
 residuals = TRUE,  
 sizeMan = 7, edge.label.cex = 1, nCharNodes = 0, nCharEdges = 0, DoNotPlot = TRUE, title = FALSE)  
plotFit1b <- mark\_sig(plotFit1, fit1)  
  
plot(plotFit1b)  
par(mar = c(5, 1, 1, 1))  
title(main = myGroup, sub = myFit,  
 col.main= "blue", col.sub = "red", cex.sub = 0.95)



I jumped over the model diagnostics section straight to the “simpler SEM”:

# simpler model:  
SEM1 <- '  
 ## Measurement model  
  
Partic\_Cont =~ V043 + V053 + V054 + V110  
Infl\_Asym =~ V111 + V112  
Inclusive =~ V048 + V049 + V051   
Deliberation =~ V070 + V093 + V094 + V096  
Facil\_Power =~ V046 + V082  
Social =~ V114 + V115 + V116 + V117 + V118 + V119 + V120  
  
 ## Structural model   
 # path: direct effect  
 Social ~ Partic\_Cont + Infl\_Asym + Inclusive + Deliberation + Facil\_Power  
'  
  
  
fit1 <- lavaan::sem(SEM1, data = myDataFacImp, ordered = colnames(myDataFacImp),  
 meanstructure = FALSE, fixed.x = FALSE, std.lv = TRUE)

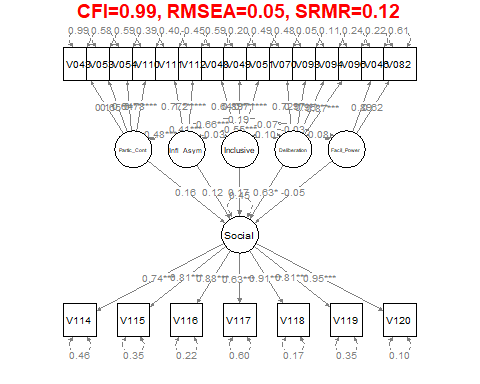
Warning: lavaan->lav\_model\_vcov():   
 The variance-covariance matrix of the estimated parameters (vcov) does not   
 appear to be positive definite! The smallest eigenvalue (= -1.699998e-15)   
 is smaller than zero. This may be a symptom that the model is not   
 identified.

Warning: lavaan->lav\_object\_post\_check():   
 some estimated ov variances are negative

myDataFactor <- myDataFacImp[ , colnames(residuals(fit1)$cov)]  
summary(fit1, standardized = TRUE)

lavaan 0.6-19 ended normally after 60 iterations  
  
 Estimator DWLS  
 Optimization method NLMINB  
 Number of model parameters 108  
  
 Number of observations 82  
  
Model Test User Model:  
 Standard Scaled  
 Test Statistic 241.114 279.416  
 Degrees of freedom 194 194  
 P-value (Chi-square) 0.012 0.000  
 Scaling correction factor 1.423  
 Shift parameter 109.961  
 simple second-order correction   
  
Parameter Estimates:  
  
 Parameterization Delta  
 Standard errors Robust.sem  
 Information Expected  
 Information saturated (h1) model Unstructured  
  
Latent Variables:  
 Estimate Std.Err z-value P(>|z|) Std.lv Std.all  
 Partic\_Cont =~   
 V043 0.110 0.135 0.819 0.413 0.110 0.110  
 V053 0.645 0.088 7.344 0.000 0.645 0.645  
 V054 0.638 0.101 6.312 0.000 0.638 0.638  
 V110 0.782 0.100 7.815 0.000 0.782 0.782  
 Infl\_Asym =~   
 V111 0.772 0.094 8.251 0.000 0.772 0.772  
 V112 1.206 0.110 11.002 0.000 1.206 1.206  
 Inclusive =~   
 V048 0.644 0.095 6.778 0.000 0.644 0.644  
 V049 0.893 0.127 7.053 0.000 0.893 0.893  
 V051 0.714 0.108 6.634 0.000 0.714 0.714  
 Deliberation =~   
 V070 0.724 0.049 14.821 0.000 0.724 0.724  
 V093 0.973 0.026 37.430 0.000 0.973 0.973  
 V094 0.946 0.017 55.355 0.000 0.946 0.946  
 V096 0.874 0.029 29.792 0.000 0.874 0.874  
 Facil\_Power =~   
 V046 0.885 0.471 1.880 0.060 0.885 0.885  
 V082 0.625 0.358 1.746 0.081 0.625 0.625  
 Social =~   
 V114 0.496 0.068 7.347 0.000 0.738 0.738  
 V115 0.544 0.068 7.959 0.000 0.808 0.808  
 V116 0.593 0.071 8.308 0.000 0.882 0.882  
 V117 0.425 0.064 6.641 0.000 0.632 0.632  
 V118 0.613 0.078 7.853 0.000 0.912 0.912  
 V119 0.543 0.071 7.659 0.000 0.807 0.807  
 V120 0.638 0.084 7.583 0.000 0.948 0.948  
  
Regressions:  
 Estimate Std.Err z-value P(>|z|) Std.lv Std.all  
 Social ~   
 Partic\_Cont 0.244 0.305 0.802 0.422 0.164 0.164  
 Infl\_Asym 0.177 0.250 0.707 0.480 0.119 0.119  
 Inclusive 0.250 0.200 1.246 0.213 0.168 0.168  
 Deliberation 0.944 0.369 2.558 0.011 0.635 0.635  
 Facil\_Power -0.077 0.236 -0.325 0.745 -0.052 -0.052  
  
Covariances:  
 Estimate Std.Err z-value P(>|z|) Std.lv Std.all  
 Partic\_Cont ~~   
 Infl\_Asym -0.475 0.124 -3.841 0.000 -0.475 -0.475  
 Inclusive 0.414 0.123 3.373 0.001 0.414 0.414  
 Deliberation 0.661 0.096 6.850 0.000 0.661 0.661  
 Facil\_Power 0.187 0.138 1.357 0.175 0.187 0.187  
 Infl\_Asym ~~   
 Inclusive -0.027 0.179 -0.151 0.880 -0.027 -0.027  
 Deliberation -0.547 0.099 -5.523 0.000 -0.547 -0.547  
 Facil\_Power -0.070 0.220 -0.318 0.750 -0.070 -0.070  
 Inclusive ~~   
 Deliberation 0.096 0.135 0.712 0.477 0.096 0.096  
 Facil\_Power -0.026 0.152 -0.170 0.865 -0.026 -0.026  
 Deliberation ~~   
 Facil\_Power 0.082 0.154 0.531 0.596 0.082 0.082

myFitMeasures <- fitMeasures(fit1, c("cfi", "rmsea", "srmr"))  
myFit <- paste(paste(c("CFI", "RMSEA", "SRMR"), round(c(myFitMeasures), 2), sep = "="), collapse = ", ")  
plotFit1 <- semPaths(fit1, what = "path", whatLabels = "stand",   
 rotation = 1, intercepts = FALSE, thresholds = FALSE,  
 sizeMan = 7, edge.label.cex = 1, nCharNodes = 0, nCharEdges = 0,   
 DoNotPlot = TRUE, title = FALSE, layout = "tree", exoVar = FALSE) # "circle2" exoVar = FALSE, exoCov = FALSE,  
plotFit1b <- mark\_sig(plotFit1, fit1)  
plot(plotFit1b)  
par(mar = c(5, 1, 1, 1))  
title(main = myFit, col.main= "red", cex.sub = 0.95)



I tried to simplify the model even further, but probably in a way that doesn’t make sense:

## CC playing around  
  
SEM\_cc <- '  
 ## Measurement model  
  
Partic\_Cont =~ V043 + V053 + V054   
Infl\_Asym =~ V111 + V112  
Social =~ V114 + V115 + V116   
  
 ## Structural model   
 # path: direct effect  
 Social ~ Partic\_Cont + Infl\_Asym   
'  
  
fit\_cc <- lavaan::sem(SEM\_cc, data = myDataFacImp, ordered = colnames(myDataFacImp),  
 meanstructure = FALSE, fixed.x = FALSE, std.lv = TRUE)

Warning: lavaan->lav\_object\_post\_check():   
 some estimated ov variances are negative

# CC:   
 fit\_cc\_resids <- resid(fit\_cc, type="cor")$cov  
range(fit\_cc\_resids)

[1] -0.2028068 0.3420947

myDataFactor <- myDataFacImp[ , colnames(residuals(fit\_cc)$cov)]  
summary(fit\_cc, standardized = TRUE)

lavaan 0.6-19 ended normally after 24 iterations  
  
 Estimator DWLS  
 Optimization method NLMINB  
 Number of model parameters 34  
  
 Number of observations 82  
  
Model Test User Model:  
 Standard Scaled  
 Test Statistic 21.629 30.254  
 Degrees of freedom 17 17  
 P-value (Chi-square) 0.199 0.025  
 Scaling correction factor 0.842  
 Shift parameter 4.562  
 simple second-order correction   
  
Parameter Estimates:  
  
 Parameterization Delta  
 Standard errors Robust.sem  
 Information Expected  
 Information saturated (h1) model Unstructured  
  
Latent Variables:  
 Estimate Std.Err z-value P(>|z|) Std.lv Std.all  
 Partic\_Cont =~   
 V043 0.237 0.163 1.460 0.144 0.237 0.237  
 V053 0.874 0.137 6.366 0.000 0.874 0.874  
 V054 0.780 0.153 5.100 0.000 0.780 0.780  
 Infl\_Asym =~   
 V111 0.569 0.206 2.757 0.006 0.569 0.569  
 V112 1.637 0.539 3.036 0.002 1.637 1.637  
 Social =~   
 V114 0.678 0.070 9.632 0.000 0.754 0.754  
 V115 0.648 0.070 9.313 0.000 0.721 0.721  
 V116 0.911 0.102 8.943 0.000 1.013 1.013  
  
Regressions:  
 Estimate Std.Err z-value P(>|z|) Std.lv Std.all  
 Social ~   
 Partic\_Cont 0.414 0.172 2.406 0.016 0.373 0.373  
 Infl\_Asym -0.171 0.178 -0.957 0.338 -0.153 -0.153  
  
Covariances:  
 Estimate Std.Err z-value P(>|z|) Std.lv Std.all  
 Partic\_Cont ~~   
 Infl\_Asym -0.257 0.141 -1.821 0.069 -0.257 -0.257  
  
  
myFitMeasures <- fitMeasures(fit\_cc, c("cfi", "rmsea", "srmr"))  
myFit <- paste(paste(c("CFI", "RMSEA", "SRMR"), round(c(myFitMeasures), 2), sep = "="), collapse = ", ")  
plotFit1 <- semPaths(fit1, what = "path", whatLabels = "stand",   
 rotation = 1, intercepts = FALSE, thresholds = FALSE,  
 sizeMan = 7, edge.label.cex = 1, nCharNodes = 0, nCharEdges = 0,   
 DoNotPlot = TRUE, title = FALSE, layout = "tree", exoVar = FALSE) # "circle2" exoVar = FALSE, exoCov = FALSE,  
plotFit1b <- mark\_sig(plotFit1, fit1)  
plot(plotFit1b)  
par(mar = c(5, 1, 1, 1))  
title(main = myFit, col.main= "red", cex.sub = 0.95)

