Gaussian Ordination

Jari Oksanen

March 28, 2024

Contents

1 Gaussian response model

The Gaussian response model for a single species and a single gradient is defined as (Fig. 1)

$$\mu_i = h \exp\left[-\frac{(x_i - u)^2}{2t^2}\right], \tag{1}$$

where μ_i is the fitted value for SU i, x_i is the gradient value, and h, u and t are the response parameters of the species: h is the maximum height of the

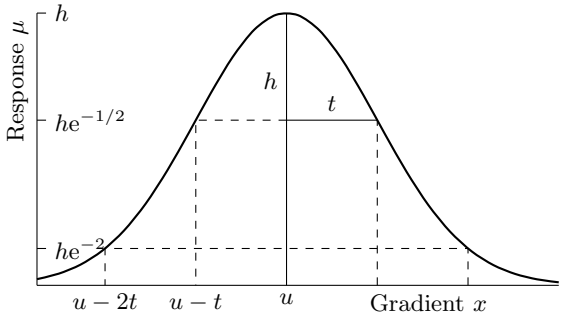


Figure 1: Gaussian response function of eq. (1).

response, u is the location of the optimum where $\mu = h$ is obtained, and t is the width or tolerance of the response.

The Gaussian response can be re-parametrized as a generalized linear model

$$g(\mu_i) = a + bx_i + cx_i^2\,, \tag{2}$$

which is equal to the Gaussian model of eq. (1) if $c < 0$. In this model, a, b and c are species parameters and $g(\cdot)$ is a link function. For Gaussian response, the link function should be $g = \log$ so that the inverse link is $g^{-1} = \exp$. In practice, binomial models are normally fitted with logistic link function which does not strictly define a Gaussian response, but is still treated similary as log link.

Polynomial form eq. (2) is a re-parametrization of the original Gaussian eq. 1, and the orginal Gaussian parameters can be found from the polynomial coefficients (provided that $c < 0$):

$$t = \sqrt{-\frac{1}{2c}} \tag{3}$$

$$u = -\frac{b}{2c} = bt^2 \tag{4}$$

$$g(h) = a - \frac{b^2}{4c} = a + 0.5b^2t^2\,. \tag{5}$$

If we scale x "in sd units" so that $t = 1$ (and hence $c = -0.5$), eq. (2) becomes

$$g(\mu_i) = a + bx_i - 0.5x_i^2\,. \tag{6}$$

With this scaling, $t = 1$, $u = b$ and $g(h) = a + 0.5b^2$. Curiously, the unimodal optima u are expressed as linear coefficients b – something we discussed with Cajo ter Braak's new electronic paper on CA. In Cajo's paper that was approximate, but here the correspondence is exact since we defined that species have unit response widths $t = 1$.

2 Gaussian Ordination

In usual model fitting with Gaussian responses, we regard gradient values x_i as known, and estimate the species parameters so that fitted values μ_i are as similar to observed values y_i as possible. In Maximum Likelihood estimation the fitted species parameters are estimated to maximize the likelihood of fitted values μ_i given data y_i and x_i. In Gaussian Ordination we select x_i and species parameters so that fitted values μ_i will maximize the likelihood function given data y_i (and the model).

We generalize eq. 6 with fixed $t = 1$ for several species j and several k gradients:

$$g(\mu_{ij}) = a_j - 0.5\sum_k x_{ki}^2 + \sum_k b_{kj}x_{ki}\,. \tag{7}$$

All terms (a_j, b_{kj} and x_{ki}) are model parameters which are estimated to maximize the likelihood of fitted values μ_{ij} given data y_{ij}. The first terms a_j define the scale or height for each species, second terms $-0.5x_{ki}^2$ scale each gradient to unit tolerance, and the last terms $b_{kj}x_{ki}$ define the relationship between the

species and the gradients. These last ones are the terms of interest: b_{kj} are the locations of optima u for species on each gradient, and x_{ki} are the gradient values.

2.1 Details of implementation

Equation (7) can be solved with non-linear minization of log-likelihood or deviance function. With a $n \times m$ data matrix and k ordination dimensions, there are m parameters a, km parameters b and kn parameters x or altogether $m + k(m + n)$ parameters which can be a large number.

There are two obvious alternative ways of estimating the model:

1. Only ordination dimensions x are found by non-linear minimization, and the species parameters a and b are found conditionally to the current iterate of x using GLM. The disadvantages of the strategy is that evaluation of the function is very expensive, because full GLM iteration is performed for every choice of x, and function may be evaluated hundreds of times in every iteration step. Further, it may be that sometimes function is trapped because species parameters adapt to the current estimates of x and prevent changing x. This is apparent as stopping because steplength is too short. Analysis of a 24×28 data set used below took nearly 400 sec in a 1.6 GHz laptop, but used only 23 iterations.

2. All parameters are found simultaneously with non-linear minization. The most obvious disadvantage is that the non-linear minimization problem can be huge. However, the same data set as above took 33 sec, although it needed 197 iterations. Large data sets can be really slow: one case I studied was a subset of 127 most common species in 398 SUs of the Mt. Field vegetation data whick in 2 dimensions gave $127+2\times(127+398) = 1177$ estimated parameters, and it took days to run this analysis in a 1.6 GHz laptop.

I have implemented both alternatives although the main development branch is based on the second alternative, and this paper only uses the second alternative. In tests, both approaches gave similar results for one dimension.

Non-linear minimization needs starting values, and because problem is large, they should be rather good. I have used site scores of detrended correspondence analysis as a starting values for x, and coefficients of the corresponding GLM (eq. 6) fit for species parameters a and b. A commonly held conjecture is that if all species have Gaussian responses with equal tolerances, DCA should be able to estimate both the locations of species optima u and gradient values x related with them. I will add an option to supply own starting values for x.

The target function to be minimized is the deviance of the model. The current version of the function uses quasi-Binomial deviance, but there will be other alternatives and this will be made a user choice. The Binomial model uses logit link function so that the models are not strictly Gaussian. Other error models to be inspected are quasi-Poisson and Normal (Gaussian distribution), both with log link.

For effective evaluation, we need derivatives of the likelihood with respect to the parameters. If these are not supplied, numerical derivatives will be used, and for each parameter this means re-evaluation of the complete loss function.

McCullagh and Nelder show that the general form of the derivate of likelihood function l with respect to the parameter p is

$$\frac{\partial l}{\partial p} = \frac{\partial l}{\partial \theta}\frac{d\theta}{d\mu}\frac{d\mu}{d\eta}\frac{\partial \eta}{\partial p} \tag{8}$$

$$= \frac{W(y-\mu)}{\mathsf{VAR}(\mu)}\frac{d\mu}{d\eta}\frac{\partial \eta}{\partial p}\,, \tag{9}$$

where W are prior weights, y are observed values (possibly scaled by W^{-1}), μ are fitted values, and η are the linear predictors, and θ is the parameter of the exponential family defining both fitted values and variances. The $\mathsf{VAR}(\mu)$ and $\frac{d\mu}{d\eta}$ functions are supplied by R as a part of defining the error distribution and link function. The partial derivatives of parameters with respect to linear predictor $\frac{\partial \eta}{\partial p}$ of eq. (2) are trivial:

$$\frac{\partial \eta}{\partial a} = 1\,, \quad \frac{\partial \eta}{\partial b} = x\,, \quad \frac{\partial \eta}{\partial x} = b - x\,. \tag{10}$$

2.2 Testable models

We have ML ordination and therefore we have all ML statistical tools available. Each model has deviance (D) which is a measure of badness of fit, and degrees of freedom of that fit. For instance, we can inspect two models with different number of extracted dimensions and see if the extra dimensions are significant. Let us inspect an alternative model with deviance D_A and k_A dimensions and a null model with deviance D_0 and k_0 dimensions, where $D_A \leq D_0$ and $k_A > k_0$. The significance of extra dimensions $k_A - k_0$ can be evaluated with F statistic

$$F_{(k_A-k_0)(m+n),mn-k_A(m+n)-m} = \frac{(D_0 - D_A)/\left[(k_A - k_0)(m+n)\right]}{D_A/(mn - k_A(m+n) - m)}\,. \tag{11}$$

A null model with no gradients ($k = 0$) will only estimate m coefficients for the average of each species, and can be used as a baseline of model comparison.

Please note that the following does not define a meanigfully testable model:

$$g(\mu_{ij}) = a_j - 0.5\sum_k x_{ij}^2\,. \tag{12}$$

This defines a model where all species have their optima u at the gradient origin $x = 0$, and superficially could be regarded as a test for an overall test that species have different optima b when compared against eq. (7). However, the location of the origin $x = 0$ is arbitrary, and tests against an arbitrary value are meaningless. The default ANOVA-style tests for GLM would use this approach, and compare model with the the linear coefficient and fixed quadratic coefficients to a model with only the fixed quadratic coefficient.

In principle it would be possible to get the estimates of standard errors of the parameters, but this is not done (yet). For this, we need to use log-likelihood as a minimized function instead of deviance (which has twice larger differences), and ask the minimizer function to return the Hessian or the matrix of the second derivatives. The estimates of the standard errors of the parameters are the square roots of inverse Hessian. These standard errors could be used to display the error variation ellipses of the ordination scores both for species and for SUs.

2.3 Constrained Gaussian Ordination

This is a work not yet done. However, the constrained version of GO looks pretty simple: Instead of freely estimating the gradients x in eq. (7), we estimate x as a function of p constraining variables z

$$x_{ki} = \sum_p \beta_{kp} z_{ip} \,, \tag{13}$$

and plug the estimated x_{ki} into eq. (7). This is actually an easier problem than unconstrained GO, because we only need to estimate kp regression coefficients (β) instead of kn gradient values, and usually $p \ll n$.

This will only give us strictly constrained scores x comparable to linear combination (LC) scores in CCA and RDA. It may be possible to get related scores that are estimates from species composition, and hence comparable to WA scores in CCA/RDA. We reverse the model and ask what are the most likely values of x giving the observed y and the current model with fixed species parameters. This is the multivariate calibration or bioindication model that I suggested in JVS 1 in 1991. However, I am not sure that this work consistently: what would it give as x if similarly applied in unconstrained GO? Consistent method should give the same x as estimated originally, but I do not believe this will happen (but will try and see).

3 Examples

I use a current and always changing version of Gaussian Ordination in these examples and data sets and functions from **vegan** package.

```
> library(GO)
> library(vegan)
```

The file `GO.R` contains main functions for analysis plus some support functions. All these are under work, and they will change, in particular in their internals, and even the user-interface is not yet stabilized. The two main analysis functions are `GO1` which implements alternate non-linear estimation of gradient values x and GLM estimation of species parameters a, b. The user interfaces are:

```
> args(GO1)

function (comm, tot = max(comm), freqlim = 5, parallel = 1, trace = TRUE,
    ...)
NULL

> args(GO)

function (comm, k = 1, tot = max(comm), freqlim = 5, family = c("poisson",
    "binomial"), far = 10, init, trace = TRUE, ...)
NULL
```

Both of these functions share some arguments:

- `comm`: The community data frame.
- `tot`: Binomial denominator.

- `freqlim`: Frequency (number of occurrences) of rarest species to be analysed. $k + 1$ parameters are fitted for each species, and therefore we need some minimal data for estimation.

Function `GO1` is so far implemented only for one dimension. Because it uses the alternating non-linear and GLM fitting, it can use parallel processing in the evaluation (`parallel` gives the number of desired parallel processes). Function `GO` can fit multidimensional models, and parameter `k` gives the number of axes. Currently `k` is limited to 4 axes, but there is no particularly compelling reason to impose this limit – I only assume that things get hairier as the number of axes increases. In addition, we can pass arguments to the minimizing function `nlm` in these functions, and we shall set higher iteration limits with `iterlim`.

`GO1` is currently only available for one dimension, and it is very slow, and therefore we use only `GO`. Let us first evaluate models for one to three dimensions:

```
> data(varespec)
> system.time(m1 <- GO(varespec, k=1, freqlim=10, tot=100, iterlim=1000))

   user  system elapsed
  0.085   0.004   0.107

> system.time(m2 <- GO(varespec, k=2, freqlim=10, tot=100, iterlim=1000))

   user  system elapsed
  0.166   0.010   0.176

> system.time(m3 <- GO(varespec, k=3, freqlim=10, tot=100, iterlim=1000))

   user  system elapsed
  0.140   0.005   0.146
```

I have written some method functions that can be used with the result:

```
> methods(class="GO")

[1] anova     calibrate plot      predict   print
see '?methods' for accessing help and source code
```

For instance, there is a `print` method so that we get a brief overview of the result (and typing the name of the object implicitly calls `print`):

```
> m1

Gaussian Ordination with 1 dimension
Call: GO(comm = varespec, k = 1, tot = 100, freqlim = 10,
iterlim = 1000)

202 iterations (converged)

Family quasipoisson
          Deviance Proportion  Df
Null     3222.0000            644
Model    1720.0000     0.5339  52
Residual 1502.0000     0.4661 592

> m2
```

```
Gaussian Ordination with 2 dimensions
Call: GO(comm = varespec, k = 2, tot = 100, freqlim = 10,
iterlim = 1000)

581 iterations (converged)

Family quasipoisson
          Deviance Proportion  Df
Null     3222.0000            644
Model    2539.0000     0.7881 104
Residual  683.0000     0.2119 540
```

```
> m3
```

```
Gaussian Ordination with 3 dimensions
Call: GO(comm = varespec, k = 3, tot = 100, freqlim = 10,
iterlim = 1000)

291 iterations (perhaps a local minimum)

Family quasipoisson
          Deviance Proportion  Df
Null     3222.0000            644
Model    2869.0000     0.8906 156
Residual  353.0000     0.1094 488
```

The `print` gives a brief overview of the result object, e.g., the astonishingly high proportions of deviance explained by each ordination. In fact, the result object contains a large number of components which are not displayed, but can be used by other functions:

```
> names(m2)
```

```
 [1] "deviance"      "null.deviance" "k"             "iterations"
 [5] "code"          "rank"          "df.residual"   "df.null"
 [9] "gradient"      "points"        "species"       "b0"
[13] "fitted"        "y"             "spdev"         "null.spdev"
[17] "family"        "call"
```

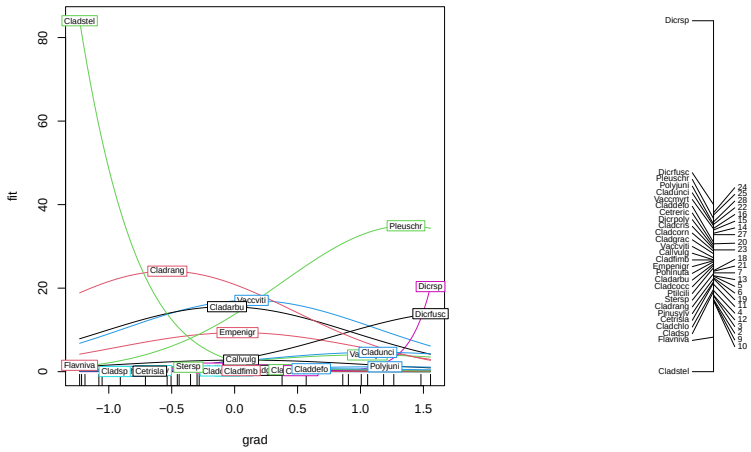
The main results are SU scores (`points`, x_{ik}) and species scores (`species`, b_{jk} which give the optima; the constants a_j are saved in item `b0`). The result object was constructed so that several standard R functions can extract the results. Moreover, **vegan** function `scores` can extract species and site scores and many standard **vegan** functions can handle the result and we do not need to write new functions or interfaces; some examples we will see here are functions are `ordiplot`, `enfvit`, `ordisurf`, `procrustes`.

I have made a specific `plot` function that displays the responses against gradients (Fig. 2).

```
> plot(m1, label = TRUE)
```

For multi-axis models this will display a line through origin with other gradients set to zero. It can optionally show the marginal model where the height of the curve would be the same on all gradients, but the default is a gradient line through the origin.

For a classical ordination plot, we can use **vegan** `ordiplot` function that is able to handle `GO` results (Fig. 2).



(a) a (b) b

Figure 2: One-dimensional Gaussian Ordination. **a** Dedicated `plot` function of `GO` showing fitted Gaussian responses against the gradient. **b** Standard **vegan** ordination plot (`ordiplot`) which for one-axis models defaults to a stacked line.

```
> ordiplot(m1)
```

Function `ordiplot` displays one-axis results with a **vegan** `linestack` function. With more axes, the default is to show ordination plots of both species and sites (Fig. 3)

```
> ordiplot(m2, type="t")
```

```
> ordiplot(m2, display="si")
```

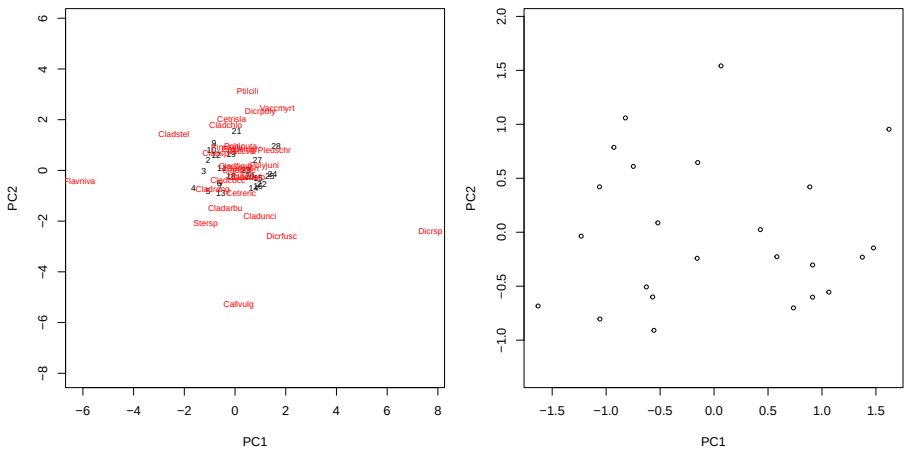
Low-dimensional solution is not a subspace in higher dimensions. We cannot add new dimensions to existing solutions, but ordination for each dimensionality must be found separately. In this respect GO and NMDS are similar. We can see this by fitting surface of one-dimensional solution onto two-dimensional solution (Fig. 4).

```
> ordisurf(m2 ~ scores(m1), bubble=3)
```

```
Family: gaussian
Link function: identity

Formula:
y ~ s(x1, x2, k = 10, bs = "tp", fx = FALSE)

Estimated degrees of freedom:
8.17  total = 9.17

REML score: -13.11604
```

```
> plot(envfit(m2 ~ scores(m1)), label="Dim1")
```

As you see, **vegan** functions `ordisurf`, `envfit` and `scores` are can handle `GO` results.



(a) (b)

Figure 3: Two-dimensional Gaussian Ordination. **a** Species optima and gradient positions of SUs. **b** Only SUs.

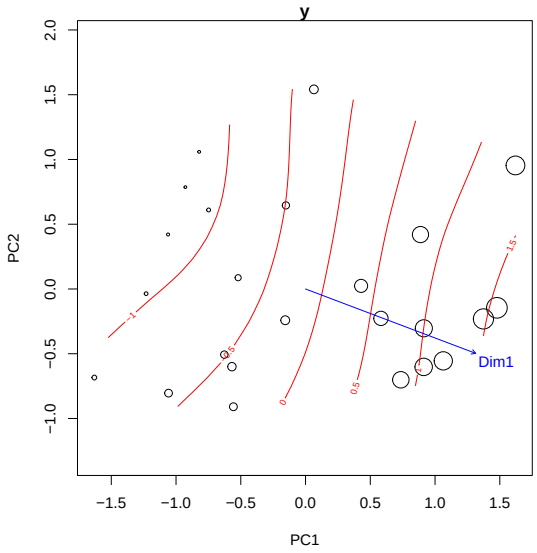
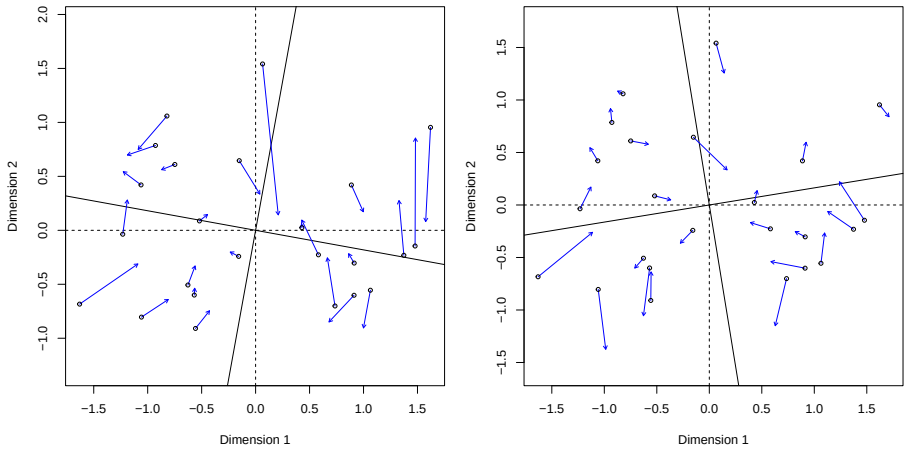


Figure 4: One-dimensional Gaussian Ordination axis fitted to a two-dimensional solution as a smooth surface and as a vector arrow. The diameter of SU circles is proportional to the scores on the one-dimensional ordination.



(a) DCA (b) NMDS

Figure 5: Procrustes rotation of 2-dimension Gaussian ordination (dots) vs (**a**) Detrended Correspondence Analysis and (**b**) Nonmetric Multidimensional Scaling (arrow heads).

3.1 Other methods

I will not launch any large scale comparison nor simulations. However, it may be good to peek how GO results compare with other methods: are they more or less similar or something completely different. I have used DCA solution as the initial values, and it has been claimed to approximate GO ML ordination – provided all species have $t = 1$, equal heights and their optima ($u = b$) are evenly distributed along the gradients. GO used only species with frequency of ≥ 10 and we shall analyse the same subset.

```
> v10 <- varespec[, colSums(varespec>0) >= 10]
> ord <- decorana(v10)
> p1 <- procrustes(m2, ord, choices = 1:2)
> p1

Call:
procrustes(X = m2, Y = ord, choices = 1:2)

Procrustes sum of squares:
5.804

> plot(p1, to.target = FALSE, main="")
```

The **vegan** `metaMDS` standardizes data by default, and this seems to have a large effect with these data. Therefore we turn off `autotransform` to analyse non-transformed data, but still use the default Bray-Curtis index.

```
> nm <- metaMDS(v10, autotransform = FALSE, trace = FALSE)
> p2 <- procrustes(m2, nm)
> p2

Call:
procrustes(X = m2, Y = nm)
```

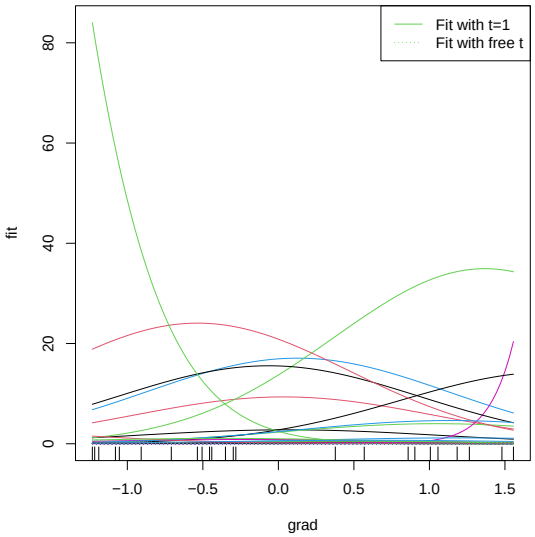


Figure 6: Gaussian ordination axis with species fitted with $t = 1$ (solid lines) and allowing freely varying t or even non-unimodal responses.

```
Procrustes sum of squares:
2.344
```

```
> plot(p2, to.target=FALSE, main="")
```

The goodness of fit is measured in the units of the target, and the analyses are comparable. NMDS is clearly much more similar to GO than DCA, although DCA was used as the starting configuration in GO.

3.2 Fixed tolerance and free shapes

I defined the species richnesses to have strictly equal tolerances $t = 1$. This does not mean that such responses really are the best ones for the gradient. In the following we fit similar second degree polynomials to all species without restricting the second degree coefficient, and plot the results over the canonical fit (Fig. 6):

```
> ## take the same subset of species as used in GO
> v10 <- varespec[, colSums(varespec>0) >= 10]
> ## the gradient
> ax <- drop(scores(m1))
> ## fit free quadratic quasibinomial GLMs
> mods <- lapply(v10, function(y) glm(cbind(y, 100-y) ~ ax + I(ax^2),
                                      family=quasibinomial))
> ## predict values
> predx <- seq(min(ax), max(ax), len=101)
> fits <- sapply(mods, function(z) predict(z, type="response",
                                           newdata=data.frame(ax=predx)))
> ## plot
> plot(m1)
```

```
> matlines(predx, fits, lty=3)
> legend("topright", c("Fit with t=1", "Fit with free t"), lty=c(1,3), col=3)
```

The widths of Gaussian responses differ from $t = 1$ although this was imposed in fitting:

```
> b <- sapply(mods, coef)
> rownames(b)

[1] "(Intercept)" "ax"          "I(ax^2)"

> sqrt(-1/2/b[3,])

 Callvulg  Empenigr  Vaccmyrt  Vaccviti  Pinusylv    Dicrsp  Dicrfusc
0.7254827 1.3882211 0.8576824 1.9786425       NaN       NaN 0.6719134
 Dicrpoly  Pleuschr  Polyjuni  Pohlnuta  Ptilcili  Cladarbu  Cladrang
      NaN 1.7164453       NaN       NaN 0.3583347 0.6572941 0.7694068
 Cladstel  Cladunci  Cladcocc  Cladcorn  Cladgrac  Cladfimb  Cladcris
      NaN 1.0294857 2.3970277 1.2404757 1.5549357 1.7469474 0.7523328
 Cladchlo    Cladsp  Cetreric  Cetrisla  Flavniva    Stersp  Claddefo
      NaN       NaN       NaN       NaN 0.1334308 0.5075505 0.6494483
```

The `NaN` cases had quadratic coefficient > 0 which define a non-unimodal response. Surprisingly, even `Cla.ste` is among those non-unimodal species, although chapter 3.4 will show it to be most important driver of the ordination axis. It seems that it would like to shoot up even more strongly than the restricted Gaussian allows. Even with unimodal species, the best fitting widths deviate from theoretical (and fitted) $t = 1$. The difference is significant, although only barely so compared to astronomically low values in all other test (see chapter 3.3):

```
> devalt <- sum(sapply(mods, deviance))
> devalt

[1] 1560.786

> deviance(m1)

[1] 1501.507

> ## we have 28 species
> dff <- 28
> dfr <- df.residual(m1)-dff
> scl <- deviance(m1)/dfr
> f <- (deviance(m1) - devalt)/dff/scl
> f

[1] -0.7952421

> pf(f, dff, dfr, lower.tail = FALSE)

[1] 1
```

3.3 Tests

We fit ML models which are special cases of GLM, and therefore we can perform all typical tests. I have packed basic ANOVA as a separate function. An overall test of the fitted model is

```
> anova(m2)
```

```
       Df Deviance Resid. Df Resid. Dev      F    Pr(>F)
NULL                    644     3221.5
Model 104   2538.9       540      682.6 19.311 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

The significances are very high (that is, P-values are low). The degrees of freedom take into account that the gradient also is estimated, but the tests still are biased. We have selected x to minimize residual deviance and maximize F statistics. I do not know yet how to develop more correct tests. Simple permutation is out of question due to slow calculation times.

The above tests gave the overall statistic for the whole multidimensional model. The packaged `anova` function allows testing a sequence of models for their difference. Comparison of `m2` against `m1` tests the significance of adding second axis to a one-axis model. Let us first see the significance of one-dimensional ordination:

```
> anova(m1)
```

```
      Df Deviance Resid. Df Resid. Dev      F    Pr(>F)
NULL                    644     3221.5
Model 52     1720       592     1501.5 13.041 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Then we can see how adding dimensions improves the results:

```
> anova(m1, m2, m3)
```

```
  Resid. Df Resid. Dev Df Deviance       F    Pr(>F)
1       592    1501.51
2       540     682.63 52   818.87 21.7986 < 2.2e-16 ***
3       488     352.54 52   330.10  8.7872 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

3.4 Contributions by species

This chapter is for those who complain that ordination tells us nothing about species.

The deviance of the model is the sum of deviances of individual species. This test is even more approximate than previous since we cannot take into account that the gradient x was estimated to minimize the sum of deviances over all species. The degrees of freedom are based only on species parameters[1] The canned test procedure for species is called `spanodev` (I know, this is an ugly name). It can be called with one model and in that case the overall model is compared against the null model. Alternatively, the function can be called with two models (but not with more models), and then these are compared against each other.

Let us see first how one dimension “explains” each species:

[1] I could take the residual d.f. per species from a residual d.f. of the complete model divided by the number of species so that the estimation of row parameters x would be divided evenly over species. This would reduce F statistics and number of d.f. and move tests to more correct direction. The implemenation of F distribution in R accepts decimal d.f.

```
> spanodev(m1)

F statistics based on (1, 21.1) degrees of freedom
           Null   Model Change        F    Pr(>F)
Callvulg 142.81 133.084   9.73   1.5456 0.2273928
Empenigr  92.74  92.272   0.47   0.1081 0.7455842
Vaccmyrt 159.44 130.776  28.66   4.6337 0.0430488 *
Vaccviti  80.33  93.935 -13.61  -3.0628 1.0000000
Pinusylv   6.31   6.148   0.16   0.5665 0.4599587
Dicrsp   176.01  25.033 150.98 127.5129 2.043e-10 ***
Dicrfusc 259.59 114.619 144.97  26.7418 3.924e-05 ***
Dicrpoly  20.32  20.581  -0.26  -0.2659 1.0000000
Pleuschr 481.50 202.917 278.58  29.0264 2.361e-05 ***
Polyjuni  38.26  33.450   4.81   3.0420 0.0956610 .
Pohlnuta   2.36   2.970  -0.61  -4.3133 1.0000000
Ptilcili  56.97  48.771   8.20   3.5537 0.0732318 .
Cladarbu 228.08 161.957  66.12   8.6317 0.0078207 **
Cladrang 291.04 160.146 130.90  17.2815 0.0004409 ***
Cladstel 905.92  59.260 846.66 302.0700 5.341e-14 ***
Cladunci 122.22  88.776  33.44   7.9647 0.0101658 *
Cladcocc   2.04   2.126  -0.09  -0.8760 1.0000000
Cladcorn   4.74   4.398   0.34   1.6250 0.2162216
Cladgrac   1.51   1.531  -0.02  -0.2997 1.0000000
Cladfimb   1.24   1.350  -0.11  -1.7602 1.0000000
Cladcris   9.17   7.378   1.79   5.1414 0.0339625 *
Cladchlo   2.54   2.380   0.16   1.4320 0.2446798
Cladsp     1.47   1.403   0.07   1.0244 0.3229115
Cetreric   5.80   6.454  -0.65  -2.1419 1.0000000
Cetrisla   4.67   4.389   0.28   1.3551 0.2573547
Flavniva  58.56  42.441  16.12   8.0312 0.0098999 **
Stersp    54.84  45.908   8.93   4.1117 0.0553759 .
Claddefo  11.03   7.053   3.97  11.9104 0.0023748 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

It is remarkable how unevenly species contribute to the model fit, and some species are actually more poorly explained by the model than the Null of constant abundance. One single species (`Cla.ste`, *Cladina stellaris*) contributes more than half of the modelled deviance. The situation is similar as in non-scaled PCA where species with large variance are the only worth explaining. Some weight equalizing trick should be developed. However, *Cladina stellaris* is an important species and probably also a keystone species which many other species depend on. Moreover, the major external driver in these data is reindeer grazing which reduces the *C. stellaris* cover drastically. We cannot say that method fails if it picks up this species.

Then we can see how adding second axis improves fits for each species:

```
> spanodev(m1, m2)

F statistics based on (1, 19.3) degrees of freedom
    Model1 Model2  Change        F    Pr(>F)
1  133.084 47.208  85.875  35.0822 9.951e-06 ***
2   92.272 50.821  41.450  15.7296 0.0008079 ***
3  130.776 41.494  89.282  41.4968 3.312e-06 ***
4   93.935 29.152  64.782  42.8573 2.664e-06 ***
```

```
5    6.148  4.495   1.654   7.0945 0.0152201 *
6   25.033 25.304  -0.271  -0.2063 1.0000000
7  114.619 52.374  62.244  22.9201 0.0001232 ***
8   20.581 11.177   9.405  16.2282 0.0007000 ***
9  202.917 85.713 117.204  26.3714 5.622e-05 ***
10  33.450 30.754   2.696   1.6904 0.2088774
11   2.970  1.914   1.055  10.6326 0.0040532 **
12  48.771 16.681  32.089  37.0990 6.945e-06 ***
13 161.957 37.414 124.543  64.1982 1.474e-07 ***
14 160.146 58.676 101.470  33.3513 1.370e-05 ***
15  59.260 55.121   4.140   1.4484 0.2433570
16  88.776 64.928  23.848   7.0835 0.0152883 *
17   2.126  1.861   0.265   2.7509 0.1133727
18   4.398  3.581   0.818   4.4045 0.0492358 *
19   1.531  1.647  -0.115  -1.3519 1.0000000
20   1.350  1.514  -0.164  -2.0896 1.0000000
21   7.378  6.769   0.609   1.7355 0.2031473
22   2.380  1.440   0.940  12.5956 0.0021040 **
23   1.403  1.434  -0.031  -0.4141 1.0000000
24   6.454  6.298   0.156   0.4774 0.4978368
25   4.389  2.333   2.056  17.0002 0.0005630 ***
26  42.441  6.402  36.038 108.5571 2.315e-09 ***
27  45.908 29.484  16.424  10.7433 0.0039009 **
28   7.053  6.643   0.410   1.1895 0.2888643
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

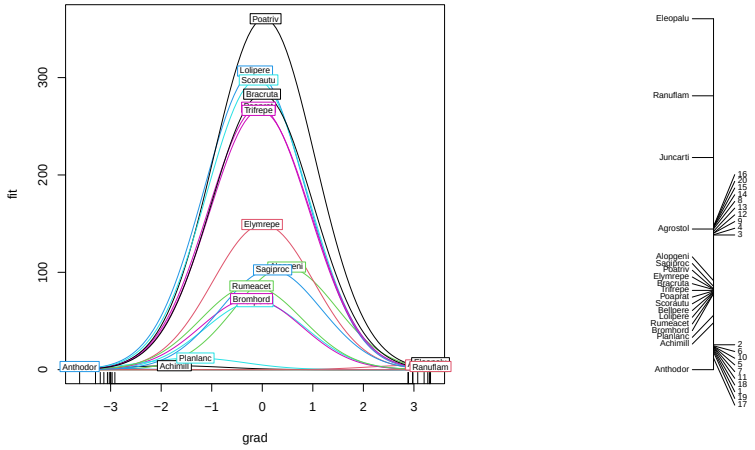
Adding second axis picks up new important species. The explanation for *Cladina stellaris* does not improve: all was explained in the first axis.

3.5 Failures

Even with limited tests, I have found cases where Gaussian Ordination seems to fail. Sometimes it fails in an informative way, sometimes mysteriously. We fit a Gaussian response model to the data, and if that model is not appropriate, ordination should fail.

One such failure case are the classic Duch Dune Meadows. Actually, one reason why I dropped developing Gaussian Ordination years ago was that I used Dune Meadows as a test case, and regarded its failure as a failure of the method. That also explains some arbitrary choices I made with the current draft version: fixing tolerances to unit scale, and starting with Binomial models with a ceiling to a response. But let us have a look at the problems:

```
> data(dune)
> mdu1 <- GO(dune, k=1, iterlim=1000)
> mdu1

Gaussian Ordination with 1 dimension
Call: GO(comm = dune, k = 1, iterlim = 1000)

69 iterations (perhaps a local minimum)

Family quasipoisson
         Deviance Proportion  Df
```



(a) (b)

Figure 7: Gaussian Ordination of Dune Meadows.

```
Null     944.7000            361
Model    464.5000     0.4917  39
Residual 480.2000     0.5083 322
```

```
> plot(mdu1, label=TRUE)
```

```
> ordiplot(mdu1)
```

The analysis did not converge, and the results are surprising (Fig. 7). SUs are polarized in two groups very far away from each other. It seems that GO thinks that there really is no continuous gradient. Most species have their optima in the empty middle region of the axis, and species are almost sure to occur in SUs that do not exist, since all SUs are somewhere else than the species optima (Fig. 7). There is no penalty of fitting any values for areas with no data. The problem was the same in earlier implementation years ago when I used quasi-Poisson models that shot up to the sky. I now switched to quasi-Binomial to avoid this, but it had no effect. The fit is evaluated at data points, and if there are no points, you can do anything with no penalty. I have feeling that most species are ubiquitous, but there is one species of dry patches (*Anthoxanthum odoratum*), and some species of wet patches (*Agrostis stolonifera*, *Juncus articulatus*, *Ranunculus flammula*, *Eleocharis palustris*). I used the term "patches" because it may be that whole plot is not dry or wet, but it has a dry (hummock) or a wet (depression) patch in addition to normal flat ground with its ubiquitous species.

I think Dune Meadows deserves to fail, and a method should fail if its strict assumptions are not appropriate. Preferably it should fail loudly, which was not the case now. Naturally, such a method is not robust, but should ML GO be robust? Isn't it Maximum Likelihood? Isn't it Gaussian? If data are not such, no ML GO should work... However, there is another case which seems to fail, and this is more problematic since here the method should have worked: GO of Mt. Field.

I intended to send you brief overview of my experiments last Sunday, but

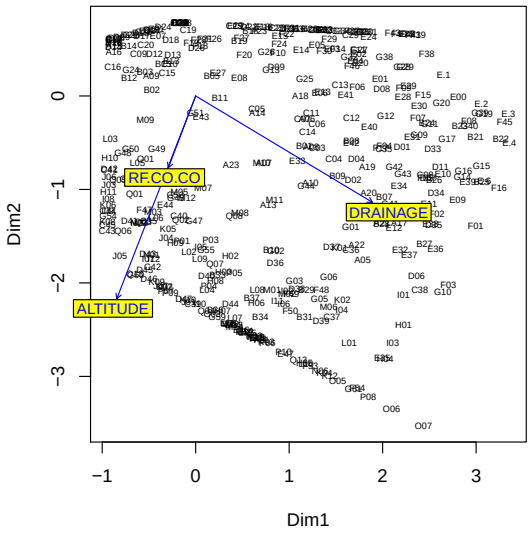


Figure 8: Two-dimensionsl Gaussian Ordination of the Mt. Field data.

then light mindedly decided to also run a GO on the full Mt. Field data set: all 398 SUs, but only 127 most common species (frequency ≥ 10). My 1.6 GHz laptop finished the analysis on Tuesday evening, and I closed the lid only when we drove from Tuusniemi to Oulu and when I rode to job on my bicycle. It took well over 50 hrs, and then it ended in a local optimum after 952 iterations. I had expected a clean result, but it looks strange (Fig. 8). Fitted vectors look OK: Altitude $R^2 = 0.8427$ vs. $R^2 = 0.8336$ and Drainage $R^2 = 0.7579$ vs. $R^2 = 0.7610$ in NMDS with the same subset of species. Moreover, Procrustes analysis hints that this has changed from the starting configuration of DCA, since Procrustes errors are smaller against NMDS than DCA. However, there are these strange straight lines that close the points in a kind of convex polygon which indicates that there is something fishy in the result. I do not know what that could be, but it does not smell success.