

ASReml-Python :: CHEAT SHEET

Getting Started

ASReml-Python is designed to fit Linear Mixed Models to data sets using complex variance models.

Installation instructions:

<https://help.vsnl.uk/asrem1-python/install>

For additional resources:

<https://help.vsnl.uk/asrem1-python>

```
import asrem1
```

or

```
from asrem1 import (asrem1, ainverse,
                    define_categorical_variables, residual_plot,
                    variogram)
```

Reading Data

Data for analysis using ASReml-Python are generally contained in a text file or a spreadsheet and are read into a **data frame** using the **Pandas** package.

```
import pandas as pd
```

```
nin89 = pd.read_csv("/path/nin89.csv",
                    sep = ",", header = 0)
```

Variety	Id	pid	raw	Rep	nloc	yield	lat	long	Row	Column
16 LANCER	1	1101	585	1	4	29.25	4.3	19.2	16	1
17 BRULE	2	1102	631	1	4	31.55	4.3	20.4	17	1
18 REDLAND	3	1103	701	1	4	35.05	4.3	21.6	18	1
19 CODY	4	1104	602	1	4	30.1	4.3	22.8	19	1
20 ARAPAHOE	5	1105	661	1	4	33.05	4.3	24	20	1
21 NE83404	6	1106	605	1	4	30.25	4.3	25.2	21	1

Character fields and numeric fields are often not treated as factors, and you must manually convert them into factors.

```
nin89 = define_categorical_variables(nin89,
                                    variables = ["Variety", "Rep", "Row", "Column"])
```

Fitting a Model

ASReml-Python uses the Wilkinson and Rogers (1973) notation. The model term A:B denotes an interaction or nested effect depending on which other terms are previously included in the model.

```
asrem1(
    response = "y",
    fixed = "Site",
    random = "Genotype + Site:Genotype",
    data = my_data)
```

Use + to add model terms

":" denotes interaction

```
asrem1(
    response = "y",
    fixed = "Site",
    random = "Site:Genotype",
    data = my_data)
```

":" denotes nested effect

An example of the `asrem1()` call for a simple randomized complete block (RCB) design using `nin89` dataframe is shown below.

```
nin89_asr = asrem1(
    response = "yield",
    fixed = "Variety",
    random = "idv(Rep, init = 10)",
    residual = "units",
    options = {"na_action":
               {"x": "include", "y": "include"}},
    predict = {"classify": "Variety"},
    vpredict = {"ICC": "V2/(V1+V2)"},
    data = nin89)
```

Set initial values (optional)

The **response**, **fixed**, **random** and **residual** terms are specified as string (use quotation marks) based on the column names in the dataset.

In **residual**, **units** is an internal experimental unit factor; and **idv()** is the homogeneous identity **variance structure** (See more at: [Variance Models](#)).

The **options** receives a dictionary with internal arguments; e.g. **"na_action"**, where: **"x": "include"** allows missing values in the design factors and **"y": "include"** response records with missing values are included.

Response Terms

The formula can specify univariate or multivariate models

```
asrem1(response = "y", ...)
```

univariate form

```
asrem1(
    response = ["y1", ..., "yk"],
    fixed = "trait + x + trait:x", ...)
```

multivariate general mean

"trait" is a reserved word

Fixed Terms

The formula can specify fixed factors, interactions and covariates.

```
asrem1(..., fixed = "Variety", ...)
```

intercept plus Variety

```
asrem1(..., fixed = "-1 + Variety", ...)
```

-1 removes the intercept

```
asrem1(..., fixed = "1", ...)
```

intercept-only

Random and Residual terms

Each random term needs to be associated to a variance model, **idv()** is assumed by default. The examples below use autoregressive matrices for both rows and columns.

```
asrem1(...,
    random = "ar1v(Column):ar1(Row)",
    residual = "idv(units)",
    ....)
```

ar1v() is a variance model

ar1() is a correlation model

In the previous example, the spatial model is defined as part of the **G** structure, and in the following example, it is defined as part of the **R** structure.

```
nin89_spt = asrem1(
    response = "yield",
    fixed = "Variety",
    random = "idv(Rep)",
    residual = "ar1v(Column):ar1(Row)",
    options = {"na_action":
               {"x": "include", "y": "include"}},
    data = nin89)
```

The operator ":" applies the direct product structure

Initial Values

The **fit_options** and **.initial_values** return a dictionary holding of the variance components

```
asrem1(...,
    fit_options = {"process_stage": "preprocess"},
    init = nin89_asr.initial_values(display = True))
```

The values can be modified:

```
init["idv(Rep)!IDV_V"]["value"] = 0.5
```

The **.fit** method runs the model with new values

```
nin89_asr.fit(initial_values = init)
```

Conditional Factors

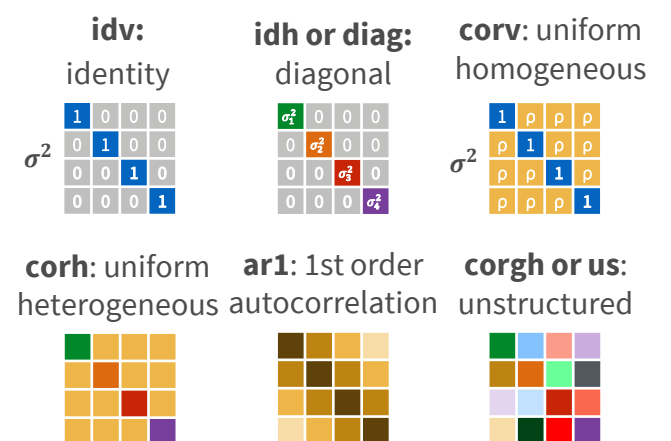
The conditional factor **at()** is used when another factor depends on specific levels on this factor.

```
asrem1(...,
    random = "at(Site, 1):Block",
    residual = "dsum(units, Site)", ...)
```

dsum() is available only for the residual formula and specifies a direct sum, one for each level of the conditioning factor.

Variance Models

ASReml-Python has a wide range of variance models that can be used to specify the variance matrix of terms in the random and residual formulae. Some of them are shown below:



The class of general variance models also includes **cor**, **corb**, **corg**, **chol**, **cholc**, **ante** and factor analytic structures (**fa**, **sfa**, **facv**, **rr**).



ASReml-Python :: CHEAT SHEET

Relationship structure

Adding a Relationship Matrix

The function **ainverse()** computes the inverse of the numerator relationship matrix from provided pedigree.

```
harvey_ped = pd.read_csv("./harvey_ped.csv")
harvey_ai = ainverse(harvey_ped)
```

	Calf	Sire	Dam		rowid	colid	vals
0	Sire_1	0	0	0	1	1	3.666667
1	Sire_2	0	0	1	2	2	3.666667
2	Sire_3	0	0	2	3	3	2.666667
3	Sire_4	0	0	3	4	4	3.666667
4	Sire_5	0	0	4	5	5	3.333333

The variance model function **vm()** allows the use of the inverse of a numerator relationship matrix or one generated from a genomics data.

```
adg_asr = asreml(
    response = "y3",
    fixed = "Line",
    random = "vm(Calf, harvey_ai)",
    residual = "units",
    data = {"data": harvey, "harvey_ai": harvey_ai})
```

variance model function

Ainv matrix

reference factor

GLMM

The inclusion of argument **family** allows for specifying other probability distributions, fitting what is referred to as a Generalized Linear Mixed Model (GLMM).

```
binor_asr = asreml(
    response = "surv",
    fixed = "Sex + Grp",
    random = "Sire",
    family = "binomial(link = logit, dispersion = 1)",
    data=binor)
```

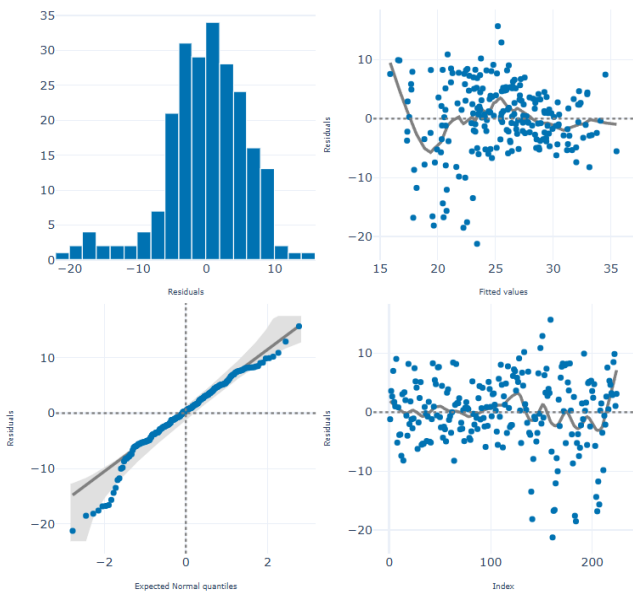
The family options are: gaussian, gamma, poisson, binomial and negative_binomial.

Plotting

Residual Plots

The **residual_plot()** function used with an **asreml** object will produce the following four residual plots:

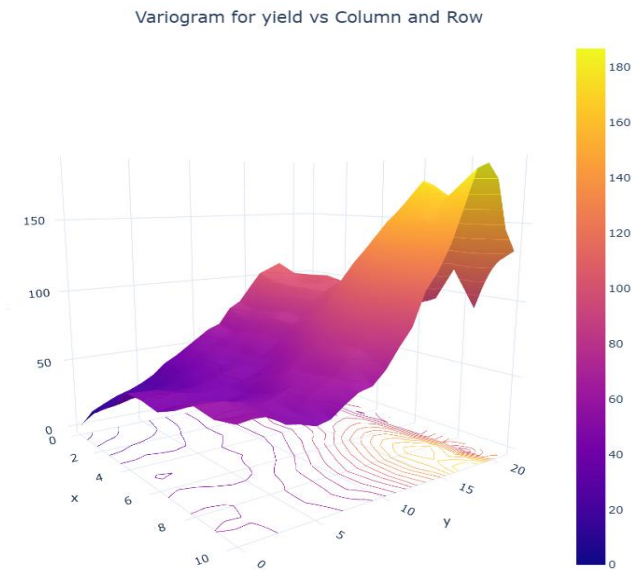
```
residual_plot(nin89_asr)
```



Variogram

For spatial models the **variogram()** will also produce a diagnostic plot based on the spatial model fitted.

```
variogram(nin89_spt)
```



Extracting Output

Summary

The **summary()** function returns a list with a variety of elements, such as:

```
nin89_asr.summary()["loglik"]
```

-454.4691

log-likelihood value

```
nin89_asr.summary()["varcomp"]
```

variance components table

	gamma	sigma	std.error	bounds	%ch
idv(units)!units!SCA_V	1.000000	49.582380	5.458841	P	0
idv(Rep)!Rep!ID_V	0.199323	9.882913	8.792661	P	0

```
nin89_asr.summary()["aic"]
```

912.9382

Akaike Information Criterion

```
nin89_asr.summary()["bic"]
```

919.1861

Bayesian Information Criterion

Solutions

The coefficients (or solutions) from solving the mixed model equations for the fixed and random parts of the model are in **.coefficients()**.

```
nin89_asr.coefficients()
```

	estimate	standard_error
Variety_ARAPAHOE	0.0000	0.000000
Variety_BRULE	-3.3625	4.979075
Variety_BUCKSKIN	-3.8750	4.979075
Variety_CENTURA	-7.7875	4.979075
Variety_CENTURK78	0.8625	4.979075

Weights

Weighted analyses are implemented by using the **weights** argument within function **asreml()**

```
asreml(..., weights = "wt", ...)
```

variate in the data frame

Wald Test

The basic tool for fixed effects inference is the **Wald** (ANOVA-like) table. The option that calculates degrees of freedom is:

```
asreml(..., options = {"wald":
    {"den_df": "numeric",
     "ss_type": "conditional"}}, ...)
```

```
nin89_asr.wald()
```

	NumDF	DenDF	F-inc	F-con	M	P-con
mu	1	3.0	242.057101	242.057101	.	0.000577
Variety	55	165.0	0.875490	0.875490	A	0.711853

Predict

The **predict** method obtains estimated or predicted means for an element or factor of interest.

```
asreml(...,
    predict = {"classify": "Variety"}, ...)
```

```
nin89_asr.predict()
```

	Variety	Pred Values	s.e.	Ecode
0	ARAPAHOE	29.4375	3.855683	E
1	BRULE	26.0750	3.855683	E
2	BUCKSKIN	25.5625	3.855683	E
3	CENTURA	21.6500	3.855683	E

Functions of Variances

The **vpredict** allows to define functions of variance components and estimate their standard error. These are represented by the strings "V1", "V2"...

```
asreml(...,
    vpredict = {"ICC": "V2/(V1+V2)"}, ...)
```

```
nin89_asr.vpredict()
```

	estimate	std.error
ICC	0.166196	0.124396

Intra-block correlation

