Analysing genetic effect models using symbolic computations

In addition to simulation and data analysis, genetic effects models can also be approached analytically with the aid of symbolic computation systems (doi:[10.1080/10705511.2024.2325122](https://doi.org/10.1080/10705511.2024.2325122), doi:[10.32614/rj-2023-090](https://doi.org/10.32614/rj-2023-090)). Here we use [SymPy](https://www.sympy.org) to encode a model symbolically and then explore it mathematically.

# import required functions  
from sympy.stats import P, E, variance, covariance, correlation, Variance, Covariance  
from sympy import Symbol, symbols, sqrt, simplify, Matrix

### Model setup

Encode additive genetic model with tagged and untagged genetic variance, plus environmental variance.

Va\_tagged = Symbol('V\_at', positive = True)  
Va\_untagged = Symbol('V\_au', positive = True)  
Ve = Symbol('V\_e', positive = True)

The phenotypic variance is therefore

Va\_tagged + Va\_untagged + Ve

Create maternal and paternal / tagged and untagged genetic effects that are normally distributed with the specified standard deviations.

# maternal and paternal tagged and untagged genetic effects  
  
# maternal tagged  
A\_mt = Normal("A\_mt", 0, sqrt(Va\_tagged))  
# maternal untagged  
A\_mu = Normal("A\_m\_u", 0, sqrt(Va\_untagged))  
# paternal tagged  
A\_pt = Normal("A\_pt", 0, sqrt(Va\_tagged))  
# paternal untagged  
A\_pu = Normal("A\_pu", 0, sqrt(Va\_untagged))

The total genetic effects for each parent are the sums of the tagged and untagged effects

A\_m = A\_mt + A\_mu  
A\_p = A\_pt + A\_pu

Offspring genetic effects have two parts: the average of the parental effects plus segregation variance (half of ).

# offspring  
# segregation variance (tagged and untagged)  
A\_ots = Normal("A\_ots", 0, sqrt(Va\_tagged/2))  
A\_ous = Normal("A\_ous", 0, sqrt(Va\_untagged/2))  
# average of parents plus segregation  
A\_ot = (A\_mt + A\_pt)/2 + A\_ots  
A\_ou = (A\_mu + A\_pu)/2 + A\_ous  
  
A\_o = A\_ot + A\_ou

Maternal, paternal, and offspring environment effects (assume uncorrelated in this model.)

# environment  
E\_m = Normal("E\_m", 0, sqrt(Ve))  
E\_p = Normal("E\_p", 0, sqrt(Ve))  
E\_o = Normal("E\_o", 0, sqrt(Ve))

Add all effects together to make phenotypes.

# phenotypes  
P\_m = A\_m + E\_m  
P\_p = A\_p + E\_p  
P\_o = A\_o + E\_o

Evaluate the phenotypic variances to check that the model was defined as expected.

variance(P\_m)

variance(P\_p)

variance(P\_o)

Finally, define additional symbols summarising variances:

Vp = Symbol('V\_p', positive = True)  
h2\_snp = Symbol('h^2\_SNP', positive = True)  
h2 = Symbol('h^2', positive = True)

### Exploring full and partial correlations

As a basic exploration, we will look at the correlation between maternal and offspring phenotypes. The covariance is:

covariance(P\_m, P\_o)

while the correlation is

mother\_offspring\_rp = correlation(P\_m, P\_o)  
mother\_offspring\_rp

If there is no untagged genetic variance (the tagged variance accounts for all of the additive genetic variance), the expected correlations is:

mother\_offspring\_rp\_Vau0 = mother\_offspring\_rp.subs(Va\_untagged, 0)  
mother\_offspring\_rp\_Vau0

Express correlation in terms of phenotypic variance and SNP heritability

mother\_offspring\_rp\_Vau0\_vp = mother\_offspring\_rp\_Vau0.subs(Va\_tagged + Ve, Vp)  
mother\_offspring\_rp\_Vau0\_vp

mother\_offspring\_rp\_Vau0\_vp.subs(Va\_tagged / Vp, h2\_snp)

Examine partial correlation between maternal and offspring phenotypes after controlling for tagged genetic effects. First, create a covariance matrix for the maternal and offspring phenotypes and tagged genetic effects:

M = [P\_m, P\_o, A\_mt, A\_ot]  
Sigma = Matrix(len(M), len(M), lambda i,j: covariance(M[i], M[j]))  
Sigma

Calculate the precision matrix as the inverse of

Omega = Sigma.inv()  
Omega

The partial correlation between the maternal and offspring phenotypes ( and ) accounting for the tagged genetic effects (, ) listed is

and are the first and second elements in , thus

def partial\_corr(Omega, i, j):  
 rho = (-Omega[i,j] / sqrt(Omega[i,i] \* Omega[j,j]))  
 return(rho)

mother\_offspring\_pheno\_partial\_corr = partial\_corr(Omega, 0, 1)  
mother\_offspring\_pheno\_partial\_corr.simplify()

and the partial correlation would be positive

mother\_offspring\_pheno\_partial\_corr > 0

If instead the tagged genetic variance is 0, then the partial correlation between maternal and offspring phenotpyes controlling for tagged genetic effects is:

mother\_offspring\_pheno\_partial\_corr.subs(Va\_untagged, 0) == 0

True

otherwise it is proportional to , which could appear to be an nurturing effect. Similarly, the correlation between the offspring phenotype and the maternal tagged genetic effect is:

correlation(P\_o, A\_mt)

while the partial correlation (controlling for maternal phenotype and offspring tagged genetic effect) is negative:

# offspring phenotype = 2nd element, maternal tagged genetic effect = 3rd element of Omega  
partial\_corr(Omega, 1, 2) < 0

### Conclusion

Using symbolic computation to explore genetic models eases the gap between writing code and mathematical analysis.

### Appendix

Instead of building the covariance matrix using a lambda constructor looping over the indices of the matrix, it can also be calculated as the expectation of the multiplication of the transposed matrix with itself:

# turn the list of variables into a matrix  
X = Matrix([M])  
# evaluating expection (E()) returns calculation in terms of variables variances  
S = E(X.T \* X)  
# invert the matrix  
O = S.inv()  
# test that this is the same as the value of Omega  
O == Omega

True