VCVs

Ian G. Brennan

4 March 2017

'split.vcv' makes two different vcv matrices split at the time 'shift.time' provided

```
release.mat <- split.vcv(tree, shift.time);
```

the first vcv matrix represents the ancient half of the tree, with max values (diagonals) equal to root height minus the 'shift.time'

```
release.mat[[1]]
```

the second vcv matrix represents the recent half of the tree, with max values (diagonals) equal to the 'shift.time'

```
release.mat[[2]]
```

*note in VCVs, 0s are a result of no observed shared ancestry, this will occur for all pairwise comparisons of individuals that are in different clades after the first (oldest) speciation event!

Let's work through this function for esimating likelihood under Brownian Motion

```
if (model == "BM") {
        k <- 2
        vcv <- vcv.phylo(tree)
        start = log(beta.start)
        lower = log(bounds[1, "beta"])
        upper = log(bounds[2, "beta"])
        foo <- function(x) {
            vv <- exp(x) * vcv
            diag(vv) <- diag(vv) + meserr^2
            mu <- phylogMean(vv, y)
            mu <- rep(mu, n)
            -dmvnorm(y, mu, vv, log = T)
        }
        o <- optim(foo, p = start, lower = lower, upper = upper,
            method = "L")
        root.state <- as.numeric(ml.root(tree = tree, model = model , y = y, meserr, params = o, shift.
        results <- list(lnl = -o$value, root.state = root.state, beta = exp(o$par))
    }
```

We can break it down into several sequential steps:
first, feed in the tree, and the parameter starting value and bounds

```
if (model == "BM") {
    k <- 2 # this is number of parameters to estimate (logLik, beta)
    vcv <- vcv.phylo(tree) # produce the variance/covariance matrix
    start = log(beta.start) # create a starting point of the optim search for beta
    lower = log(bounds[1, "beta"]) # create a lower bound for the optim search for beta
    upper = log(bounds[2, "beta"]) # create an upper bound for the optim search for beta
```

now we have to make the function to calculate the logLik and beta

```
    foo <- function(x) {
```

transform the tree by multiplying the var/covar values by an estimated beta (exp(beta)), optimized in the 'optim' step

```
        vv <- exp(x) * vcv
```

if applicable, take into account measurement error

```
        diag(vv) <- diag(vv) + meserr^2
```

estimate the phylogenetic mean for each trait value using the vv matrix and empirical data, then repeat for each species observation (ntaxa). this is the optimum trait value, or long-term mean.

```
        mu <- phylogMean(vv, y)
        mu <- rep(mu, n)
```

calculate the multivariate normal density from your (y) data, (mu) phylogenetic means, and (vv) transformed tree matrix

```
        -dmvnorm(y, mu, vv, log = T)
    }
```

finally, we'll optimize the beta using our starting value, and upper and lower bounds

```
    o <- optim(foo, p = start, lower = lower, upper = upper,
        method = "L")
```

then estimate the root value, whch is dependent upon your model and best estimated parameters

```
    root.state <- as.numeric(ml.root(tree = tree, model = model ,
                                     y = y, meserr, params = o, shift.time));
```

your results, including your logLik, root state estimate, and rate (beta = sigma.sq)

```
    results <- list(lnl = -o$value, root.state = root.state, beta = exp(o$par))
}
```

If we want to use more complex models, we'll want to include an OU process
This requires transforming the VCV matrix using an OU model

```
ouMatrix <- function (vcvMatrix, alpha)
{
    vcvDiag <- diag(vcvMatrix) # grab out the diagonals from the VCV
    diagi <- matrix(vcvDiag, nrow = length(vcvDiag), ncol = length(vcvDiag)) # set diagonals of taxon '
    diagj <- matrix(vcvDiag, nrow = length(vcvDiag), ncol = length(vcvDiag), # set diagonals of taxon '
        byrow = T)
    # these correspond to the pairwise comparison of taxa in the BM process
    Tij = diagi + diagj - (2 * vcvMatrix) # the raw maximum distance among pairs
    vcvRescaled = (1/(2 * alpha)) * exp(-alpha * Tij) * (1 -
        exp(-2 * alpha * vcvMatrix)) #
    return(vcvRescaled)
}
```

$$cov(X_i, X_j) = \frac{\sigma^2}{2\alpha} e^{-2\alpha(T - t_{ij})}(1 - e^{-2\alpha t_{ij}}) \qquad (A6)$$

Figure 1: from Cooper et al. (2016) "A cautionary note...". BJLS