Calculation of the Hessian matrix of branch lengths

Calculating the Hessian on an alignment of one partition

Let $\hat{\mathbf{b}} = (\hat{b}_i)$, $i = 1, ..., 2s-3$ be the vector of maximum likelihood estimates (MLEs) of branch lengths on an unrooted phylogeny of s species, calculated on an alignment with n_T sites. Let $\hat{\mathbf{g}}_i = (\hat{g}_{i,j})$ be the vector of length n_T of first derivatives of the log-likelihood function calculated at each site in the alignment and evaluated at $\hat{b}_i$. Note that if $0 < \hat{b}_i < \infty$, then $\sum_j \hat{g}_{i,j} = 0$. Let $\hat{\mathbf{G}}$ be a matrix with $2s-3$ rows and n_T columns, where the i-th row is $\hat{\mathbf{g}}_i$. Following Seo et al. (2004), the Hessian matrix (the matrix of second derivatives of the log-likelihood function) evaluated at the branch length MLEs is

$$\hat{\mathbf{H}} = -\hat{\mathbf{G}}\hat{\mathbf{G}}^{\mathrm{T}}. \tag{1}$$

Usually alignments are compressed into site-patterns. Let $\mathbf{n} = (n_k)$ be the vector of site-pattern counts, i.e., n_k is the number of times site-pattern k is observed. Thus, $\sum_k n_k = n_T$. Now let $\hat{\mathbf{g}}_i$ and $\hat{\mathbf{G}}$ be the corresponding gradient vector and matrix calculated on the site patterns. Let $\mathbf{N} = \mathrm{diag}(\mathbf{n})$ be the diagonal matrix of site pattern counts. The Hessian is $\hat{\mathbf{H}} = -\hat{\mathbf{G}}\mathbf{N}\hat{\mathbf{G}}^{\mathrm{T}}$.

Note Eq. (1) assumes all MLEs of branch lengths are different from zero (see Appendix in dos Reis and Yang, 2011), thus if any $\hat{b}_i = 0$, then Eq. (1) will not produce the right Hessian. Currently, MCMCtree uses Eq. (1) to calculate the Hessian, and the program prints a warning if any $\hat{b}_i = 0$. However, we note that, in any case, the Taylor approximation is still reasonably good becasue for zero-length branches the gradient term dominates the Taylor expansion, and the constribuiton of the Hessian is small.

Yang (2000) gives a method to calculate the gradient and the diagonal of the Hessian analytically, and this method is currently implemented in IQ-TREE. Thus, a good alternative is to use (1) to calculate the Hessian and then replace the diagonal with the analytic estimates.

Calculating the Hessian on an alignment with many partitions

Suppose the alignment is divided in several partitions, with some species possibly missing in some partitions. Let s_j be the number of species in partition j. Currently, MCMCtree calculates the likelihood for each partition independently, assuming unlinked branch lengths across partitions. Thus, each partition would have $2s_j - 3$ estimated branch lengths, requiring its own gradient and Hessian calculation.

Some notes on methods to calculate the Hessian

Let $\ell(\mathbf{x} \mid \mathbf{b}, \theta)$ be the log-likelihood of a sequence alignment $\mathbf{x}$ on an unrooted phylogeny of s species, given the branch lengths, $\mathbf{b}$. The log-likelihood can be approximated by its Taylor expansion

$$\ell(\mathbf{x} \mid \mathbf{b}) \approx \ell(\mathbf{x} \mid \hat{\mathbf{b}}) + (\mathbf{b} - \hat{\mathbf{b}})^{\mathrm{T}}\hat{\mathbf{g}} + \frac{1}{2}(\mathbf{b} - \hat{\mathbf{b}})^{\mathrm{T}}\hat{\mathbf{H}}(\mathbf{b} - \hat{\mathbf{b}}),$$

where $\hat{\mathbf{b}}$ are the MLEs of the branch lengths, $\hat{\mathbf{g}} = (\hat{g}_i = \partial\ell/\partial b_i|_{b_i=\hat{b}_i})$ is the gradient vector of first derivatives evaluated at the MLEs, and $\hat{\mathbf{H}} = (\hat{h}_{ij} = \partial^2\ell/(\partial b_i \partial b_j)|_{b_i=\hat{b}_i, b_j=\hat{b}_j})$ is the Hessian matrix of second derivatives evaluated at the MLEs. The Hessian is of size $(2s-3) \times (2s-3)$. The approximate likelihood can be calculated much quicker than the exact likelihood, which requires Felsenstein prunning algorithm. Thus, the approximate likelihood is a good choice for Bayesian MCMC inference of species divergence times (note the approximation works on a fixed tree topology, and thus the equation above cannot be used if the tree topology changes during Bayesian inference). The approximate method for Bayesian time estimation was proposed by Thorne et al. (1998), with Seo et al. (2004) and dos Reis and Yang (2011) providing further improvements. To use the approximation one first needs to calculate $\hat{\mathbf{b}}$ and $\hat{\mathbf{H}}$. Although calculation of $\hat{\mathbf{g}}$ numerically is easy, calculation of $\hat{\mathbf{H}}$ is usually more difficult. A few methods to obtain $\hat{\mathbf{H}}$ are listed below.

1. **Analytic**. For phylogenies of two species under simple substitution models, $\hat{\mathbf{H}}$ may be available analytically, for example, in the Jukes-Cantor model (see Eqs. (8) and (9) in dos Reis and Yang, 2011). The advantage of this method is that $\hat{\mathbf{H}}$ is calculated without error, but its disadvantage is that it is not available for real data analyses in which complex models and phylogenies of many species are common.

2. **Numerical differentiation.** Standard numerical methods can be used to obtain $\hat{\mathbf{H}}$. For example, Eqs. (18) and (19) in the appendix of dos Reis and Yang (2011) give a common formula using the difference method. This was the original method used by Thorne et al. (1998) and also used in older versions of PAML. The disadvantage of this method is that, sometimes, it is numerically unstable.

3. **Outer product of scores (OPS).** This is the method suggested in the appendix of Seo et al. (2004, Eq. (16)) to avoid the numerical probles in the difference method. The OPS method is discussed in Porter (2002, see fifth-paragraph in p. 432). Let $\hat{\mathbf{g}}_j$ be the gradient vector of length $2s-3$ evaluated at the MLEs for the j-th site in the alignment (note this is different to $\hat{\mathbf{g}}_i$ in the previous section, which is of length n_T). The OPS estimator is

$$\hat{\mathbf{H}} \approx -\sum_{j=1}^{n_T} \hat{\mathbf{g}}_j \hat{\mathbf{g}}_j^{\mathrm{T}}.$$

   This is the same as Eq. (16) in Seo et al (2004). Porter (2002) gives the same equation but muliplied by n_T^{-1}. The vector crossproduct results in a $(2s-3) \times (2s-3)$ matrix for site j, $\hat{\mathbf{G}}_j = \hat{\mathbf{g}}_j \hat{\mathbf{g}}_j^{\mathrm{T}}$, and thus the Hessian is the sum over these site matrices, $\hat{\mathbf{H}} \approx -\sum_{j=1}^{n_T} \hat{\mathbf{G}}_j$. Note that we can re-arrange terms to re-write the OPS estimator as $\hat{\mathbf{H}} = -\hat{\mathbf{G}}\hat{\mathbf{G}}^{\mathrm{T}}$ as explained in the previous section. Both formulae are equivalent. Porter (2002) give two further methods to estimate $\hat{\mathbf{H}}$ but it is not clear if these are useful in our case. The advantage of the OPS method is that it is numerically stable and also fast to calculate. Its disadvantage is it does not provide the right values of $\hat{h}_{ij}$ if $\hat{b}_i = 0$ or $\hat{b}_j = 0$. However this error does not appear important in divergence time estimation (see appendix in dos Reis and Yang, 2011). This is the method currently implemented in PAML and it is widely used for Bayesian MCMC divergence time estimation.

4. **Hybrid analytic/numerical.** It may be possible to calculate $\hat{\mathbf{g}}$ and the diagonal elements of the Hessian, $\hat{h}_{ii}$, analytically using the re-rooting method of Yang (2000). Then the off-diagonal elements can be calculated by using the difference method on $\hat{\mathbf{g}}$, as given in Eq. (20) of dos Reis and Yang (2011). This method is probably the best, as it should be numerically stable (the difference formula is only applied once on the gradient instead of twice as in the full difference method) and it should provide the correct entries, $\hat{h}_{ij}$, in all cases (i.e. without errors for $\hat{b}_i = 0$). This method has not been implemented in PAML, because the default optimisation algorithm in PAML is the simultaneous optimisation method and not the branch-by-branch (or re-rooting) method of Yang (2000).

5. **Brute-force boostrap.** Asymptotically, the variance-covariance matrix of the branch-length MLEs is $\hat{\mathbf{C}} = -\hat{\mathbf{H}}^{-1}$. Thus, one could generate bootstrap replicates of an alignment $\mathbf{x}^*$, and then calculate the branch-length boostrapped MLEs, $\hat{\mathbf{b}}^*$. Say, we do 100 boostrap replicates, then we get 100 vectors $\hat{\mathbf{b}}^*$. We then get the bootstrap estimate of the covariance $\hat{\mathbf{C}}^*$. Then the Hessian is $\hat{\mathbf{H}} \approx -\hat{\mathbf{C}}^{*-1}$. This method has just been proposed by Wang and Luo (2023) so that they could use the sophisticated substituiton models in IQ-tree. The advantage of this method is that it can be virtually applied to any substitution model available in any ML phylogenetics software. Its disadvantage is that it is computationally expensive and it is not clear how it deals with zero-length branches.

Worked example: A worked example, using a two-species phylogeny under K80 subsitution model, is provided in the `test.R` file (function definitions are in the `k80-hessian.R` file). Note the two-species K80 case has two parameters to be estimated, a single branch length, *b*, and the transition-transversion ratio, κ. The example calculates the corresponding 2×2 Hessian using various meethods. In divergence time estimation, we ignore the contribution of substitution model parameters, such as κ, to the Hessian, but we use it here for academic purposes as the methods above apply to all MLEs. Table 1 shows calculations of $\hat{\mathbf{H}}$ under three methods for this worked example: (i) The first method is to use the numerical difference method directly to get $\hat{\mathbf{H}}$. In `test.R`, the code that does this is in line 24:

```
h <- -numDeriv::hessian(k80.optim.wrap, c(0.1045798, 30.8364620),
  ns=84, nv=6, n=948)
```

(ii) The second method is to calculate $\hat{\mathbf{g}}$ analytically (this is available for the K80 model with two species). Then we use the formula from the previous section with site patterns. In `test.R`, the code that does this is in lines 17 to 20:

```
# calculate gradient on site patterns analytically:
g.m <- k80.gradient(d=mle["d"], k=mle["k"], n=3, ns=1, nv=1)
# hessian by Seo's method:
h1 <- -g.m %*% diag(nn) %*% t(g.m)
```

(iii) The third method calculates the site-gradients by the difference method, and then uses the formula by Seo et al. (2004, also Potter 2002). Note Seo's formula has to be corrected for the site-patterns. In `test.R`, the code that does this is in lines 31 to 39:

```
# calculate gradient on site patterns using numerical optimisation:
g0 <- -numDeriv::grad(k80.optim.wrap, mle, ns=0, nv=0, n=n0)
# on p0 (conserved sites)
g1 <- -numDeriv::grad(k80.optim.wrap, mle, ns=n1, nv=0, n=n1)
# on p1 (ts differences)
g2 <- -numDeriv::grad(k80.optim.wrap, mle, ns=0, nv=n2, n=n2)
# on p2 (tv differences)
G <- cbind(g0, g1, g2)
apply(G, 1, sum) # [1]  4.797104e-08 -4.385377e-12
# This is the Hessian matrix calculated by Seo's method
h2 <- -(tcrossprod(g0/sqrt(n0)) + tcrossprod(g1/sqrt(n1)) +
  tcrossprod(g2/sqrt(n2)))
```

All methods give essentially the same result, within acceptable numerical error (Table 1).

Table 1: Hessian calculations for the two-species K80 case. The data are a pairwise sequence alignment with 948 sites, of which 84 are transitions and 6 are transversion differences. This is the example in p. 9 of Yang (2014).

(i)	Difference method on $\hat{\mathbf{H}}$	
	b	κ
b	−7452.704108	0.314043049
κ	0.314043	−0.005821949
(ii)	$\hat{\mathbf{g}}$ analytic, $\hat{\mathbf{H}} = -\hat{\mathbf{G}}\mathbf{N}\hat{\mathbf{G}}^{\mathrm{T}}$	
	b	κ
b	−7453.2905734	0.314032612
κ	0.3140326	−0.005821896
(iii)	Difference method on $\hat{\mathbf{g}}_j$, $\hat{\mathbf{H}} \approx -\sum_{j=1}^{n_T} \hat{\mathbf{g}}_j \hat{\mathbf{g}}_j^{\mathrm{T}}$	
	b	κ
b	−7453.2905735	0.314032613
κ	0.3140326	−0.005821896

References

- dos Reis M, and Yang Z. (2011) MBE, 28: 2161–2172.
- Porter J. (2002) J. Buss. Econ. Stat., 20: 431–440.
- Seo TK, Kishino H, and Thorne JL. (2004) MBE, 27: 1201–1213.
- Thorne JL, Kishino H, and Painter S. (1998) MBE, 15: 1647–1657.
- Wang S, and Luo H. (2023) bioRxiv, 2023.06.18.545440.
- Yang Z. (2000) JME, 51: 423–432.
- Yang Z. (2014) Molecular Evolution: A Statistical Approach. OUP.