

Burnin, the convergence diagnostic and user trees

What does burnin really do?

It is no wonder that MrBayes users find **burnin** a confusing topic. The burnin option is used with three different commands and has different meanings in those commands, but all have to do with the issue of convergence.

The most familiar use is with the **sumt** and **sump** commands. In those commands burnin is the number of samples that are ignored when summarizing the trees or parameters respectively. For example, `sumt burnin = 2500;` means that in summarizing the trees in order to calculate a consensus tree the first 2500 trees will be ignored. That setting means that you expect MrBayes to have converged on a set of statistically indistinguishable trees by the time it has sampled 2500 trees.

But how do you know if MrBayes has actually converged at that point? In version 3 MrBayes introduced a new tool, the convergence diagnostic, to evaluate whether the program has converged. The convergence diagnostic reports the standard deviation of the split frequencies, and its use is controlled by the **mcmc** (or **mcmcp**) command.

By default the convergence diagnostic is printed to the screen every 1000 generations, and is based on the last 75% of the samples up to that point. At generation 1000 the diagnostic (std dev of split frequencies) is calculated based on the last 750 generations, and at generation 10,000 it is reported based on the last 7500 generations. As MrBayes converges the diagnostic approaches zero. The MrBayes development team has told me that you can be very confident of convergence when the diagnostic is <0.01 , but that a diagnostic <0.05 is acceptable.

I have suggested setting burnin for **sumt** and **sump** to a number that is 25% of the total number of trees; i.e. if `ngen = 1,000,000` and you sample every 100 generations there will be 10,000 trees at the end of the run so I suggest that you would set `burnin = 2500`. In the default situation that would mean that if the convergence diagnostic was <0.01 at the end of the run, convergence had occurred by the time the first 25% of the trees had been estimated. If you want to use some other setting for the `sumt` and/or `sump` burnin setting you will want to change the convergence diagnostic option under the `mcmc` command.

There are four options under the `mcmc` command that deal with the convergence diagnostic: **Relburnin** determines whether the burnin is a fraction of the samples or is an absolute number of samples. The default is `relburnin = yes`.

Burninfrac determines the fraction of samples that will be ignored when the convergence diagnostic is calculated, and only applies when **relburnin** = yes. You should set burninfrac to the fraction of trees that is burnin under the `sumt` and `sump` commands. The default is `burninfrac = 0.25`.

Burnin determines the number of samples that will be ignored when the convergence diagnostic is calculated, and only applies when **relburnin** = no. The default is `burnin = 0`.

Diagnfreq determines how often the convergence diagnostic is printed to the screen. The default `diagnfreq = 1000` means that it is printed every 1000 samples.

Suppose that I have a large alignment, over 100 sequences, in which the sequences are very diverse so that I suspect that it may take a long time to converge. I want my consensus tree to be based on 10,000 post-convergence trees. I guess that it might take 1,000,000 generations to converge so I want a total of 2,000,000 generations (or 20,000 samples, sampling every 100 generations), and I want to discard the first 10,000 or 50% of my trees.

```
mcmc ngen=2000000 samplefreq = 100 printfreq = 1000 diagnfreq =  
10000 relburnin = yes burninfrac = 0.5;
```

```
sumt burnin = 10000;
```

With such a long run I set it to print to the screen only every 1000 generations and to calculate the convergence diagnostic only every 10,000 generations. Printing slows things down and with such a long run it is worth saving the print time.

By the way, to see explanations of all of the commands under **mcmc** start MrBayes and type `help mcmc`. You will have to scroll up to see all of the commands. Under **relburnin** it says that the default is `relburnin = no`, but that is incorrect. To see the default settings look at the bottom of the help screen to see that `relburnin` is really set to `yes`.

Using the convergence diagnostic to end the run

Sometimes you set a large number of generations only to see convergence occur early in the run. After that you watch with increasing annoyance as the run continues adding pointless generations. Suppose that you specified `ngen = 1000000`, only to see that by generation 200,000 the convergence diagnostic is <0.01 . There was no way to know in advance that the program would converge so early, but it would sure be nice to be able to set it up so that the run would stop when convergence has occurred.

Actually, you can do that by setting two **mcmc** (or **mcmcp**) options.

Stoprule ends the run before the number of generations set by **ngen** when the convergence diagnostic falls below **stopval**. Default is `stopval = no`.

Stopval sets the critical value of the convergence diagnostic and only applies when **stoprule** = `yes`.

Suppose I have set:

```
mcmc ngen=1000000 samplefreq = 100 printfreq = 1000 diagnfreq =  
10000 relburnin = yes burninfrac = 0.25 stoprule=yes stopval=0.01;  
sumt burnin = 2500;
```

Now, if the convergence diagnostic <0.01 at generation 200,000 the run will stop at that point. **But** there will only be 2000 trees at that point, and I set `sumt` to discard the first 2500 trees. Oops!

Actually, there is no problem. Although the run is done, if I have used the command `set autoclose=no` MrBayes will still be running and I can simply type:

```
Sumt burnin = 500
```

at the MrBayes prompt. Of course, the actual number of trees will correspond to the `burninfrac` set in `mcmc`.

For maximum flexibility set `autoclose` to `no` as suggested in the MrBayes templates, use the `stoprule`, and don't include a `sumt` command in the MrBayes block. That way, if convergence occurs before the number of generations set in `ngen` MrBayes will ask you if you want to do more generations, you will say "no", and you will enter the `sumt` command manually. On the other hand, if convergence does not occur before the number of generations set in `ngen`, when MrBayes asks if you want more generations say yes and add more generations. The `stoprule` will still apply. If you use this approach you won't have to worry about guessing the number of generations that will be needed for convergence

What if I can't get MrBayes to converge?

If the convergence diagnostic is <0.05 I generally don't worry about it. However, if it is >0.05 there are a couple of things to try.

[Change the temperature](#)

The temperature affects how often MrBayes switches between chains. The temperature is set as an **mcmc** (or **mcmcpr**) option. The default is `temp=0.2`. Increasing the temperature increases switch attempts, but decreases the likelihood of the switch being accepted; decreasing temp decreases the switch frequency but increases the likelihood of a switch being accepted. I have had good luck **both** with `temp=0.25` and `temp = 0.15`. I wouldn't go much outside of that range.

Use a good tree as a starting point

By default MrBayes starts from a random tree as it seeks better and better trees. With some data sets (alignments) that does not work very well. The alternative is to provide MrBayes with a pretty good tree as a starting point.

1. Use the alignment in Phylip format to make an ML tree with `phymml` or `phymml-mlrt` as described in Chapter 7. Don't have it calculate branch supports either by bootstrapping or by the various aLRT possibilities.
2. Open the resulting tree file in MEGA, set it to show topology only and not to display branch lengths, then export it as a Newick file.*
3. In the MrBayes block, on a line just before the `mcmc` command insert the command `usertree =`, then copy the tree from the exported Newick file and paste it after the "=" sign. This defines the user tree as the ML tree. For some reason MrBayes hangs if the user tree includes branch lengths, so be sure to eliminate branch lengths as suggested in step 2.
4. In the **mcmc** command add the option `startingtree=usertree`.

* Macintosh users who don't have access to MEGA may find it convenient to open the ML tree in MacClade and then to export it as a Phylip file without branch lengths.

By using the ML tree as a starting point MrBayes starts pretty close to the final tree, and it wanders around among very similar trees. As a result it is likely to converge reasonably soon.