

Tutorial 6

TNT - tipos de caracteres e topologias de consenso

BIZ0433 - INFERÊNCIA FILOGENÉTICA: FILOSOFIA, MÉTODO E APLICAÇÕES.

Conteúdo

Objetivo	94
6.1 Tipos de caracteres	95
6.2 Árvores de consenso	101
6.2.1 Consenso estrito	102
6.2.2 Consenso semi-estrito	102
6.2.3 Consenso de Maioria	102
6.2.4 Estabilidade do consenso	103
6.3 Referências	106

Objetivo

O primeiro objetivo deste tutorial é apresentar como caracteres são, ou podem ser tratados em análises filogenéticas. Neste componente do tutorial iremos explorar os impactos do ordenamento de caracteres em inferência filogenética e aplicação de custos arbitrários para séries de transformação. O segundo objetivo deste tutorial é apresentar algumas técnicas de consenso e como computá-las operacionalmente em TNT. Por fim, é apresentado um protocolo para averiguar estabilidade de topologias de consenso em espaços de árvores complexos. Os arquivos associados a este tutorial estão disponíveis no [GitHub](https://github.com/fplmarques/cladistica). Você baixar todos os tutoriais com o seguinte comando:

```
svn checkout https://github.com/fplmarques/cladistica/trunk/tutorials/
```

6.1 Tipos de caracteres

Dentro do contexto de homologia estática de caracteres (*sensu* Wheeler [1]), no qual caracteres e seus respectivos estados de caráter (*i.e.*, séries de transformação) são postulados a priori, existem 3 classes de caracteres: aditivos (ordenados), não-aditivos (não-ordenados) e caracteres de Sankoff (matrizes de custo). Caracteres aditivos [2] são aqueles em que o custo de transformação é determinado pela diferença entre o índice de cada estado no qual cada índice sucessivo representa um incremento de proposições de homologias mais restritivo [3]. Considere por exemplo a Figura 6.1A. Assumindo que a raiz desta transformação é o estado “0”, a aquisição do estado “2” assume que a transformação passou pelo estado “1”, ou seja, “0” → “1” → “2”. Por outro lado, séries de transformações não-aditivas [4], o custo de cada transformação é constante entre qualquer par de estados de caráter (veja Figura 6.1B). Neste caso, a aquisição do estado “2” assume apenas uma transformação independente do estado plesiomórfico do qual este derivou (*i.e.*, “0”, “1”, ou “3”; Figura 6.1B).

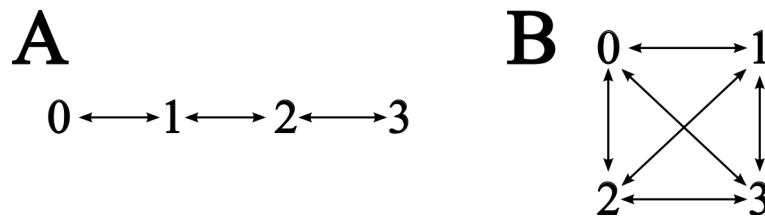


Figura 6.1: Tipos de caracteres: aditivo e não-aditivo. **A**, caráter aditivo; **B**, caráter não-aditivo.

Não há consenso na literatura sobre ordenar ou não séries de transformações de caracteres multi-estados (seja [5–7]). Nixon & Carpenter [7:8] argumenta:

Additive characters are just more explicit, compound hypotheses of homology. The fact that nonadditive codings tend to produce shorter trees does not a priori make them better. Throwing away characters, or lying, can also produce shorter trees. Nonadditive codings are better only when they are better justified (or more defensible) in terms of homology assessment. In fact, a nonadditive multistate character implies ambiguity in our understanding of the homology among the states, because it is not generally possible that all allowed transformations are simultaneously true.

Os argumentos apresentados por Nixon & Carpenter [7] sugerem que é necessário explicitar os critérios que levam o investigador a considerar determinada série de transformação de forma aditiva. Isso porque seguindo o raciocínio desses autores, o ordenamento revela maior conhecimento sobre as noções de homologia putativa entre os estados. No entanto, Hauser & Presch [5] verificaram que a grande maioria dos estudos que consideravam séries de transformações ordenadas não apresentava nenhum critério para determinar a ordem dos estados de caracteres. Tendências morfológicas, sequências ontogenéticas (raramente disponíveis), similaridade entre estados, entre outros, podem ser utilizadas para justificar ordem de estados de caráter [mas veja 5] –, nenhuma delas sem assumir premissas cuja necessidade fica a cargo

do pesquisador. Considere que o ordenamento de estados de caráter representa uma hipótese específica sobre a relação evolutiva entre os estados de caráter.

Vejamos como o TNT permite implementar estes conceitos. A definição de tipos de caracteres em TNT é feita pelo comando “ccode”:

```
tnt*>help ccode
CCODE
! re-sets ccode to the one defined in the data file
Other than that, sets character codes. Specifiers are:
+ make following character(s) additive
- " " " non-additive
[ " " " active
] " " " inactive
( " " " Sankoff
) " " " non-Sankoff
/N apply weight N to following character(s)
=N apply N additional steps to following character(s)
```

Vejamos como podemos implementar caracteres aditivos e não aditivos em TNT. Considere a seguinte matrix:

```
xread
7 6
taxon_A 0000000
taxon_B 1000000
taxon_C 2000011
taxon_D 3001112
taxon_E 4111212
taxon_F 5111322
;
```

Considere os seguintes comandos e seus respectivos efeitos:

- i. “ccode + 0”: Faz com que TNT considere o caráter 0 aditivo.
- ii. “cc + 0 4.6”: Faz com que TNT considere os caracteres 0, 4, 5 e 6 aditivos¹.
- iii. “cc -.”: Faz com que TNT considere todos caracteres não-aditivos.
- iv. “cc”: Faz com que TNT exiba a codificação de cada caráter, por exemplo:

Ccode

¹Você pode abreviar esse comando utilizando simplesmente “cc”

+ [/1 0 - [/1 1 - [/1 2 - [/1 3 + [/1 4
 + [/1 5 + [/1 6 ;

Neste exemplo acima, os caracteres 0, 4, 5 e 6 serão considerados aditivos (*i.e.*, identificados pelo símbolo “+”). Observe também que a notação “[/1”, comum a todos os caracteres, indica que todos os caracteres estão sendo considerados e possuem peso 1.

Exercício 6.1

Neste exercício, iremos utilizar a matriz contida no arquivo `exemplo_1a.tnt` para implementar os conceitos descritos acima.

- i. Faça uma análise cladística da matrix em TNT do arquivo `exemplo_1a.tnt` e responda:
 - a. A topologia depende do ordenamento de algum caráter? Justifique.

- ii. Faça uma análise cladística da matriz em TNT do arquivo `exemplo_1b.tnt` e responda:
 - a. Qual caráter quando considerado aditivo reduz o número de topologias igualmente parcimoniosa?

- b. Existe algum caráter cujo ordenamento (*i.e.*, “0” → “1” → “2” ...) pode ser defendido utilizando o critério de parcimônia? Justifique.

- c. Supondo que você tenha uma justificativa para ordenar o caráter que reduz o número de topologias, implemente a ordenação deste caráter e salve no arquivo

`exemplo_1b_recons.txt` a reconstrução do caráter 2² e responda: Quantas reconstruções o TNT postula para esse caráter?

O ordenamento de caracteres não precisa ser necessariamente linear (*i.e.*, “0” → “1” → “2” → “3” → “4”) como nos exemplos acima. Considere, por exemplo, a Figura 6.2. Nela, as relações entre os estados de caráter 0–4 é apresentada de forma ramificada. Assumir essa estrutura hierárquica entre os estados de caráter não é diferente do ordenamento linear desta série de transformação. No entanto, esta relação entre os estados requer uma implementação distinta em TNT. A matriz à direita na Figura 6.2 expressa os custos associados entre cada uma das transformações possíveis. Por exemplo, a transformação “1” → “3” tem o custo de 3 transformações (*i.e.*, “1” → “0” → “2” → “3”).

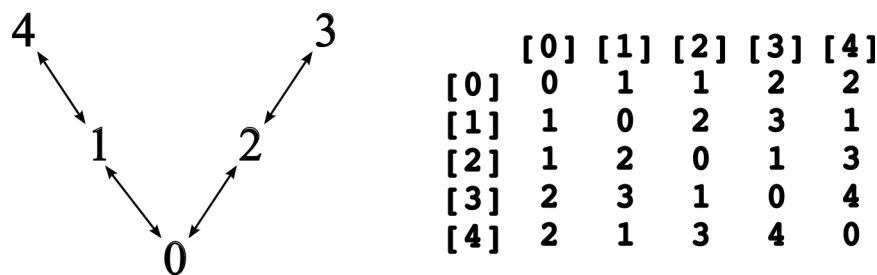


Figura 6.2: Tipos de caracteres: matriz de transformação.

Esta matriz define os custos de transformação do que chamamos caracteres de Sankoff [8]. Matrizes de Sankoff são implementadas para caracteres para os quais assume-se custos arbitrários de transformação, via de regra associados à uma ou mais premissas sobre a evolução destes caracteres. Embora Sankoff [8] tivesse desenvolvido este procedimento de otimização para estudar evolução macromolecular, matrizes de Sankoff podem ser aplicadas a qualquer série de transformação – mesmo binária.

Em TNT, a implementação de matrizes de Sankoff requer os seguintes passos:

1. Definição da matriz de Sankoff:

A definição de matrizes de Sankoff é feita pelo comando `smatrix` do TNT obedecendo a seguinte sintaxe:

```
smatrix =S (xxxx) ...costs... ;
```

onde “S” é um número entre 0-31 e indexa internamente a matriz no TNT; “xxxx” é o nome atribuído à matriz (opcional) e “costs” são os custos de cada transformação. Os custos são definidos considerando a direção da transformação, como por exemplo 0>4 2 que atribui custo 2 para a transformação “0” → “4”, ou estabelecendo custos simétricos, como por exemplo 0/2 1 que atribui custo 1 para as transformações “0” → “2” ou “2” → “0”.

²numeração de acordo com TNT. Veja Tutorial 5 item 5.2 para detalhes de como verificar reconstruções em TNT.

2. Aplicação da matriz de Sankoff:

Uma vez definida a matriz de Sankoff, ela deve ser aplicada ao caráter desejado, ou caracteres utilizando o comando “`smatrix +S N`” ou “`smatrix +xxxx N`”, onde “N” é o caráter para o qual se quer aplicar a matriz de Sankoff.

3. Implementação matriz de Sankoff em `ccode`:

Finalmente, é necessário habilitar o caráter de Sankoff no TNT utilizando o comando “`ccode (N`”.

Exercício 6.2

Neste exercício, você deverá criar uma matriz de Sankoff e implementar esta função de custo na análise filogenética usando TNT. **No entanto, antes de executar esse exercício examine as últimas linhas do arquivo `exemplo_2.tnt`.**

- i. Construa uma matriz de Sankoff para a série de transformação ordenada ilustrada na Figura 6.3.

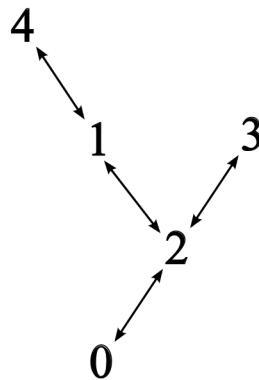


Figura 6.3: Série de transformação ordenada não-linear.

	[0]	[1]	[2]	[3]	[4]
[0]	-				
[1]		-			
[2]			-		
[3]				-	
[4]					-

- ii. Com base na sua matriz de Sankoff, defina a `smatrix` abaixo:

```
smatrix =0 (minha_matriz) ...
```

iii. Execute uma busca no TNT com o arquivo `exemplo_2.tnt` e responda:

A implementação da matriz de Sankoff teve algum impacto na reconstrução do caráter 2? Justifique.

Matrizes de Sankoff são muito utilizadas em análise de dados moleculares – em concordância com sua concepção inicial [8]. A Figura 6.4 representa duas classes de transformações (*i.e.*, transições e transversões) de caracteres genotípicos – sequências nucleotídicas – comumente utilizadas em análises filogenéticas de dados moleculares. A premissa associada ao custo diferencial entre essas duas classes de transformações reside na expectativa de que transversões (*i.e.*, purinas \longleftrightarrow pirimidinas) tem impacto bioquímico mais acentuado nas moléculas e, conseqüentemente, estão sob maiores restrições de transformação do que transições (*i.e.*, purinas \longleftrightarrow purinas ou pirimidinas \longleftrightarrow pirimidinas). Sequências nucleotídicas são sempre consideradas caracteres não-aditivos e via de regra são submetidos à matrizes de Sankoff.

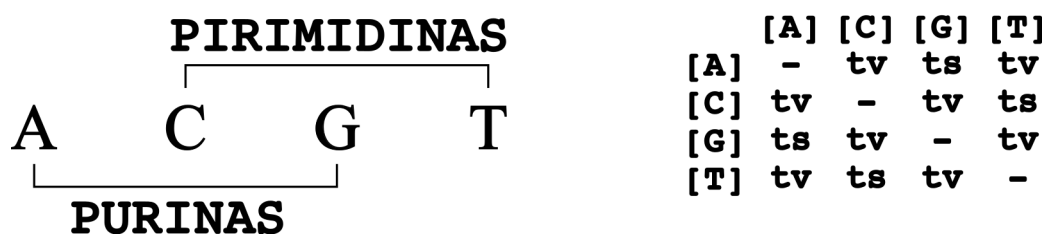


Figura 6.4: Classes de transformação de caracteres genotípicos: transições e transversões

Exercício 6.3

Considere os dados na matriz do arquivo `exemplo_3.tnt` e execute as seguintes tarefas:

i. Faça uma análise filogenética destes dados e responda:

a. Qual é a topologia recuperada e qual é o seu custo (*i.e.*, número de transformações)?

ii. Abaixo defina a matriz de Sankoff no qual você atribuirá peso 2 para transversões e peso 1 para transições:


```
smatrix =0 C/A _ G/A _ G/C _ T/A _ T/C _ T/G _;
```

iii. Modifique o arquivo `exemplo_3.tnt` de modo que todos os caracteres sejam analisados de acordo com sua matriz de Sankoff, reanalise esses dados e responda:

a. A implementação desta nova função de custo modificou seus resultados? Explique.

6.2 Árvores de consenso

Métodos de consenso apresentam um sumário de informações contidas em um conjunto de topologias. Fundamentalmente, topologias de consenso é o resultado da aplicação de uma série de regras e/ou algoritmos sob um conjunto de topologias que resulta em uma única árvore com o mesmo conjunto de terminais [9]. De acordo com Bryant [9], há uma série de controvérsias relacionadas à topologias de consenso; no entanto estas estão mais relacionadas com as interpretações sobre as topologias de consenso do que com os métodos usados para gerá-las [veja, 10].

Há algumas generalidade sobre topologias de consenso que devem ser consideradas. A primeira delas é que topologias de consenso são sumários de informação e o número de transformações nestas topologias é via de regra subótima [11, 12] – portanto, **otimizações em topologias de consenso não devem ser consideradas**. Outra propriedade comum à topologias de consenso, especialmente relacionadas com o sumário de informações filogenéticas provenientes de fontes de dados distintas, é que em alguns casos a topologia de consenso difere do resultado que seria obtido considerando a análise simultânea destas bases de dados (seja Barrett *et al.* [10]). Portanto, considere que a maioria das técnicas de consenso foram concebidas como ferramentas de representação e não para **inferência filogenética** propriamente dita (mas veja Holder *et al.* [13]).

Neste componente do tutorial iremos explorar três tipos de consenso, aqueles mais frequentemente utilizados em análises filogenéticas. Essa exposição breve sobre o tópico tem mais caráter operacional do que fomentar as discussões sobre o uso de métodos de consenso em inferência filogenética.

6.2.1 CONSENSO ESTRITO

O consenso estrito [14], como o nome sugere, é o mais simples – e o mais frequente na literatura. Esta técnica de consenso produz uma topologia que é o sumário de todos os componentes (*i.e.*, clados) que estão presentes em **todas** as topologias fundamentais.

6.2.2 CONSENSO SEMI-ESTRITO

O consenso semi-estrito [15], também conhecido como *compatible components*, é menos restritivo do que o consenso estrito. Neste método de consenso a topologia final contém todos os componentes presentes nas topologias fundamentais **com a inclusão daqueles que não são contraditórios entre si**. Colocado de outra forma, enquanto que no cálculo do consenso estrito um determinado componente só será representado se estiver necessariamente em todas as topologias fundamentais, no consenso semi-estrito ele deve estar presente em pelo menos uma delas e não ser contradito por nenhum outro componente das demais topologias que fazem parte do conjunto.

6.2.3 CONSENSO DE MAIORIA

O consenso de maioria (*i.e.*, *majority-rule*; Margush & McMorris [16]) baseia-se na frequência dos clados presentes nas topologias fundamentais. Margush & McMorris [16] define este tipo de consenso como sendo topologias de consenso M_l no qual o parâmetro l define a porcentagem mínima da frequência dos componentes que deverão estar presentes na topologia de consenso. Observe que se l é 100% você obtém o consenso estrito das topologias fundamentais. Este tipo de consenso é utilizado em análises de suporte e de inferência filogenética que utiliza como critério de otimização probabilidades posteriores [*i.e.*, análises bayesianas; veja 13].

Exercício 6.4

Neste exercício iremos explorar as propriedades destes métodos de consenso utilizando as topologias existentes no arquivo `exemplo_4.tnt`.

- i. Verifique os clados em cada uma das três topologias.
- ii. Calcule o consenso estrito utilizando o comando “`ne`”.
- iii. Represente a topologia de consenso no espaço abaixo e compare novamente com as topologias fundamentais.

- iv. Calcule o consenso semi-estrito utilizando o comando “`comcomp`”.

- v. Represente a topologia de consenso no espaço abaixo e compare novamente com as topologias fundamentais.

- vi. Como é a resolução desta topologia de consenso em relação à topologias fundamentais?

- vii. Calcule o consenso de maioria utilizando o comando “majority = 50”.

- viii. Represente a topologia no espaço abaixo e compare novamente com as topologias fundamentais.

- ix. Qual observação você faria ao comparar as topologias fundamentais com a de consenso de maioria?

6.2.4 ESTABILIDADE DO CONSENSO

Independente da técnica de consenso que você deseja utilizar para calcular topologias de consenso, é importante que você tenha uma boa amostra de topologias do seu espaço de árvore – principalmente quando este apresenta relativa complexidade. Neste último componente sobre o assunto irei sugerir um protocolo para verificar se, potencialmente, você obteve estabilidade em sua topologia de consenso. Considere que nem sempre a inspeção visual é uma forma imediata de observar se houve ou não mudança entre uma topologia de consenso e outra à medida em que você compila topologias com o mesmo custo. Considere por exemplo os arquivos `consenso_*.tre`. Suponha que esses arquivos foram obtidos com buscas incrementalmente mais agressivas e que à cada uma delas um número maior de topologias foi amostrado. Se você listar de forma longa esses arquivos (*i.e.*, com o comando `ls -l`), você obterá:

```
-rw-rw-r- 1 alan alan 84 Apr  7 20:29 consenso_1.tre
```

```
-rw-rw-r- 1 alan alan 82 Apr  7 20:33 consenso_2.tre
-rw-rw-r- 1 alan alan 80 Apr  7 20:33 consenso_3.tre
-rw-rw-r- 1 alan alan 80 Apr  7 20:33 consenso_4.tre
```

A primeira dica refere-se ao tamanho desses arquivos. Observe que o arquivo `consenso_1.tre` possui 84 bites, o `consenso_2.tre` 82 e os demais 80. O número decremental de bites sugere que os arquivos possuem um número menor de caracteres, ou seja, parênteses que são usados para definir grupos! Inspeção o conteúdo destes arquivos.

No caso da sugestão acima, você deve considerar que dois arquivos podem possuir o mesmo número de bites e, no entanto, suas topologias podem ser distintas. Um outra forma de verificar, ou comparar essas topologias de forma mais segura, é utilizar o comando “`diff`” – um comando interno de LINUX/UNIX. Se você executar em um terminal:

```
$ diff -q -s consenso_1.tre consenso_2.tre
```

você deverá obter:

```
Files consenso_1.tre and consenso_2.tre differ
```

Por outro lado, se você executar em um terminal:

```
$ diff -q -s consenso_3.tre consenso_4.tre
```

você deverá obter:

```
Files consenso_3.tre and consenso_4.tre are identical
```

Exercício 6.5

Neste exercício você irá explorar estabilidade de consensos. Você deverá fazer algumas análises em TNT, compilar os dados na Tabela 6.1 e identificar a partir de que momento destas análises você obteve a estabilidade de sua topologia de consenso.

Considere os seguintes comandos de TNT e suas respectivas ações:

```
log log_run.txt;
xmu: hold 10 rep 50 ratchet 5 drift 5 fuse 10;
xmu;
ne*;
tchoose/;
tsave* busca_1.tre;
save;
tsave/;
log/;
```

Esta sequência de comandos abre um arquivo de *log* chamado `log_run.txt`, executa uma busca usando novas tecnologias em TNT para um determinado arquivo de entrada, calcula o consenso estrito e faz com que a topologia de consenso seja inserida no *buffer* de memória do TNT, seleciona a última topologia e descarta as demais (no caso a topologia de consenso é mantida), abre um arquivo para salvar topologias chamado `busca_1.tre`, salva a topologia e fecha os arquivos de árvore e de *log*.

Neste exercício você deverá estabilizar o consenso as topologias encontradas na matriz `zilla.tnt` em 10 buscas. Os números de topologias encontradas em cada uma dessas análises deverá ser sempre superior à encontrada na análise anterior. Você deverá preencher a tabela abaixo (Tabela 6.1) e comparar os arquivos que julgar necessário com o comando `diff -q -s [arquivo1] [arquivo2]`.

Tabela 6.1: Buscas heurísticas e estabilidade de consenso em TNT

Busca	Parâmetros de Busca	# Topologias retidas	Tamanho do Arquivo
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			

Observações de comparação:

6.3 Referências

1. Wheeler, W. C. 2001. Homology and the optimization of DNA sequence data. *Cladistics* **17**: S3–S11.
2. Farris, S. 1970. Methods for computing Wagner trees. *Systematic Zoology* **19**: 83–92.
3. Wheeler, W. C. 2012. Systematics: a course of lectures. Malaysia: Wiley-Blackwell, 2012. 426.
4. Fitch, W. M. 1971. Toward defining the course of evolution: Minimum change for a specific tree topology. *Systematic Zoology* **10**: 406–416.
5. Hauser, D. L. & Presch, W. 1991. The effect of ordered characters on phylogenetic reconstruction. *Cladistics* **7**: 243–265.
6. Slowinski, J. B. 1993. "Unordered" versus "ordered" characters. *Systematic Biology* **42**: 155–165.
7. Nixon, K. C. & Carpenter, J. M. 2011. On homology. *Cladistics* **27**: 1–10.
8. Sankoff, D. 1975. Minimal mutation trees of sequence. *SIAM Journal on Applied Mathematics* **28**: 35–42.
9. Bryant, D. em *DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Volume 61* eds. Janowitz, M. F.; Lapointe, F. J.; McMorris, F. R.; Mirkin, B & Roberts, F. S. Providence, Rhode Island: American Mathematical Society, 2003.
10. Barrett, M.; Donoghue, M. J. & Sober, E. 1991. Against consensus. *Systematic Zoology* **40**: 486–493.
11. Miyamoto, M. M. 1985. Consensus cladograms and general classifications. *Cladistics* **1**: 186–186.
12. Carpenter, J. M. 1988. Choosing among equally parsimonious cladograms. *Cladistics* **4**: 291–296.
13. Holder, M. T.; Sukumaran, J. & Lewis, P. O. 2008. A justification for reporting the Majority-rule consensus tree in Bayesian Phylogenetics. *Systematic Biology* **57**: 814–821.
14. Rohlf, F. J. 1982. Consensus indices for comparing classifications. *Mathematical Biociences* **59**: 131–144.
15. Bremer, K. 1990. Combinable component consensus. *Cladistics* **6**: 369–372.
16. Margush, T & McMorris, F. R. 1981. Consensus n-trees. *Bulletin of Mathematical Biology* **2**: 239–244.