Integer Linear Programming Models for Multiple Sequence Alignment

Ignasi Andreu Godall

Scientific director: Gabriel Valiente (UPC)

Motivation: Finding an optimal solution to a multiple sequence alignment problem instance for more than two sequences is a hard optimization problem that is prohibitively computationally expensive, even for a few sequences of moderate length. Consequently, research on multiple sequence alignment has mainly focused on heuristic methods. However, recent advances in solvers for integer linear programming have made it possible to find exact, optimal solutions to multiple sequence alignment problem instances for several sequences of moderate length.

Results: In this paper, we review existing integer linear programming models for multiple sequence alignment, and present a new model based on the longest common subsequence problem. While existing models define $O(n\ell^2)$ binary variables and $O(n\ell^4)$ constraints to align n sequences of length at most ℓ, our new model defines $O(n\ell^2)$ binary variables and also $O(n\ell)$ integer variables, but only $O(n\ell^2)$ constraints. This allows for finding an exact solution to the multiple sequence alignment problem for several sequences of moderate length in a reasonable amount of time on a personal computer.

Supplementary information: Supplementary data are available in an appendix.

1 Introduction

Multiple sequence alignment (MSA) is a popular research area in bioinformatics that consists in arranging more than two biological sequences, usually DNA, RNA, or protein sequences, in such a way that the resulting arrangement can be used to identify regions of similarity, which may result from functional, structural, or evolutionary relationships among the sequences. Thus, multiple sequence alignment plays an essential role and has many applications in fields such phylogenetics, structural bioinformatics, and comparative and functional genomics. For example, Figure 1 shows a multiple alignment of 14 proteins that are known to bind sugar. A pairwise alignment might already reveal some commonalities and differences among the sequences, but the multiple alignment shows that all the molecules share eight sulfur-containing cysteines, which are known to form four disulfide bridges [3].

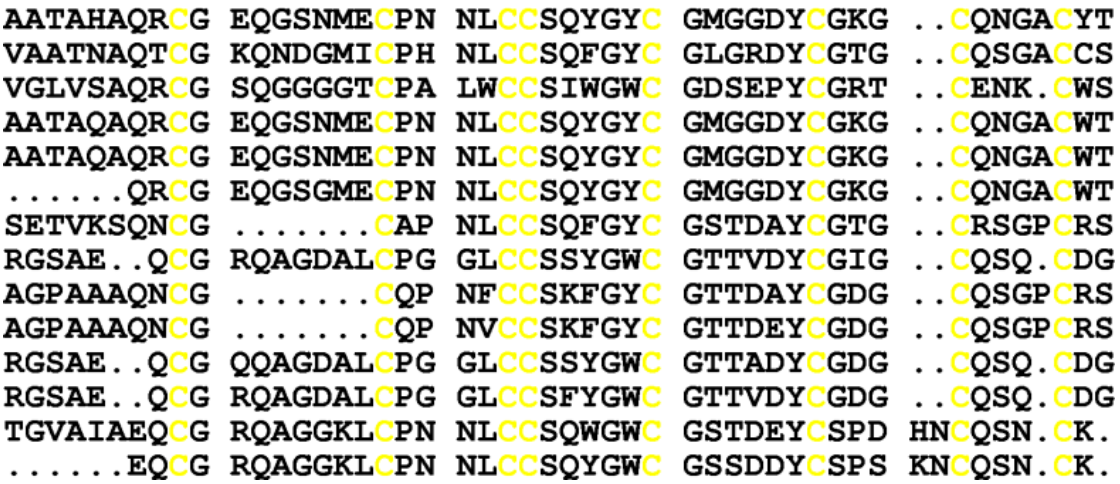

Figure 1: A multiple sequence alignment of N-acetylglucosamine-binding protein sequences highlighting conserved cysteines, adapted from [3].

From a computational point of view, multiple sequence alignment is a hard optimization problem that involves the consideration of all different possible pairwise alignments in order to find an optimal one. Given a set S of sequences, a multiple sequence alignment of S is obtained by inserting gaps ('-') to represent insertions and deletions into the original sequences, such that all resulting sequences are of the same length and no column consists of gaps only. Thus, the original sequences can be recovered by removing all gaps from the aligned sequences.

The most common scoring methods used to decide where to insert gaps and how to align the sequences are: sum of pairs (computing all the costs among all the pairs of amino acids in the aligned columns), entropy (minimizing the number of different amino acids in each aligned column), and consistency (maximizing the alignment using a list of constraints).

Given n sequences of length at most ℓ, their multiple alignment using n-dimensional dynamic programming takes $O(\ell^n)$ time, which comes as no surprise, as multiple sequence alignment is known to be an NP-hard problem [16]. Therefore, many software tools that use heuristics have emerged, such as Clustal Omega [13], MUSCLE [7], and T-Coffee [11], which do not guarantee an optimal solution to the multiple sequence alignment problem but that often produce good enough alignments.

However, recent advances in mathematical methods and software tools for integer linear programming make it possible to find optimal solutions to many interesting problems that arise in computational biology and bioinformatics. In fact, several problems related to multiple sequence alignment have already been studied using integer linear programming models [1, 2, 3, 10]. See [9, 12] for a broader overview of integer linear programming models in computational biology and bioinformatics.

2 Methods

We developed two ILP models for the multiple sequence alignment problem, both of which are based on the longest common subsequence (LCS) problem, a classical computer science problem that has applications in computational linguistics, bioinformatics, and many other fields.

In particular, in bioinformatics, LCS is used for sequence alignment in comparative genomics, for phylogenetic construction and analysis, for rapid search in huge biological sequences, for compression and efficient storage of the rapidly expanding genomic data sets, and for re-sequencing a set of strings given a target string, which is an important step in efficient genome assembly [4].

Given a set S of sequences, each of which is a string of symbols from a finite alphabet Σ, an LCS for S is a string of symbols from Σ of maximum length that is contained as a subsequence in each string S. For example, string QCGCPCCSGCGDYCCC is a longest common subsequence of the N-acetylglucosamine-binding protein sequences in Figure 2.

2.1 First ILP Model for Multiple Sequence Alignment

The starting point towards an ILP model for multiple sequence alignment is an ILP model for the longest common subsequence problem, adapted from [6]. It takes the following input parameters: n, the number of sequences, and $S = \{s_1, \ldots, s_n\}$, a set of n sequences.

Let us define an additional parameter, $m_{i,j,p,q}$, for each pair of consecutive sequences s_i and s_j, position $p = 1, \ldots, |s_i|$, and position $q = 1, \ldots, |s_j|$, as follows:

$$m_{i,j,p,q} = \begin{cases} 1, & \text{if } s_i[p] = s_j[q] \\ 0, & \text{otherwise} \end{cases}$$

Let us also define a binary variable $x_{i,j,p,q}$ for each pair of consecutive sequences s_i and s_j, position $p = 1, \ldots, |s_i|$, and position $q = 1, \ldots, |s_j|$, where $x_{i,j,p,q} = 1$ if the symbols $s_i[p]$ and $s_j[q]$ are part of a *longest common subsequence*, and $x_{i,j,p,q} = 0$ otherwise. Variables x encode thus a solution to an LCS problem instance, as enforced by the following linear constraints:

1. Symbols $s_i[p]$ and $s_j[q]$ can be part of a longest common subsequence ($x_{i,j,p,q} = 1$) if and only if they are identical ($m_{i,j,p,q} = 1$). For $i = 1, \ldots, n-1$, $j = i+1$, $p = 1, \ldots, |s_i|$, $q = 1, \ldots, |s_j|$,

$$x_{i,j,p,q} \leqslant m_{i,j,p,q}$$

2. A symbol $s_i[p]$ can be aligned with at most one symbol $s_j[q]$. For $i = 1, \ldots, n-1$, $j = i+1$, $q = 1, \ldots, |s_j|$,

$$\sum_{p=1}^{|s_i|} x_{i,j,p,q} \leqslant 1$$

3. A symbol $s_j[q]$ can be aligned with at most one symbol $s_i[p]$. For $i = 1, \ldots, n-1$, $j = i+1$, $p = 1, \ldots, |s_i|$,

$$\sum_{q=1}^{|s_j|} x_{i,j,p,q} \leqslant 1$$

4. A symbol $s_i[p]$ that is aligned with a symbol $s_j[q]$ must also be aligned with some

symbol $s_k[r]$. For each three consecutive sequences s_i, s_j and s_k, with $i = 1,\ldots,n-2$, $j = i+1$, $k = j+1$,

$$x_{i,j,p,q} \leqslant \sum_{r=1}^{|s_k|} x_{j,k,q,r}$$

where $p = 1,\ldots,|s_i|$, $q = 1,\ldots,|s_j|$, and

$$x_{j,k,q,r} \leqslant \sum_{p=1}^{|s_i|} x_{i,j,p,q}$$

where $q = 1,\ldots,|s_j|$, $r = 1,\ldots,|s_k|$.

5. The symbols that comprise a solution must appear in the same order in all the sequences. This means, there cannot be "crosses" between these symbols in any pair of consecutive sequences. For $i = 1,\ldots,n-1$, $j = i+1$, $p_1,q_1 = 1,\ldots,|s_i|$, $p_2,q_2 = 1,\ldots,|s_j|$,

   $$(p_1 > p_2 \wedge q_1 < q_2) \vee (p_1 < p_2 \wedge q_1 > q_2)$$

   implies

   $$x_{i,j,p_1,q_1} + x_{i,j,p_2,q_2} \leqslant 1$$

The goal of this ILP model for the longest common subsequence problem is to maximize the length of the common subsequence. Since the symbols of a common subsequence are represented by one values of the variable x, the objective function is to maximize its sum over all pairs of consecutive sequences s_i and s_j, positions p of sequence s_i, and positions q of sequence s_j,

$$\max \sum_{\substack{i=1,\ldots,n-1 \\ j=i+1 \\ p=1,\ldots,|s_i| \\ q=1,\ldots,|s_j|}} x_{i,j,p,q}$$

In order to extend this first ILP model for the longest common subsequence problem to solve the multiple sequence alignment problem, let us redefine parameter m in terms of nucleotide or amino acid substitution matrices. While the m parameter only allows for the alignment of identical symbols, the new m parameter will take values from a given substitution matrix. That is,

$$m_{i,j,p,q} = \text{substitution\_matrix}[s_i[p], s_j[q]]$$

Let us also define a gap penalty parameter, which can be set according to the particular substitution matrix being used, and that allows for reducing the number of gaps introduced in a multiple sequence alignment. For example, in the case of the Blosum62 amino acid substitution matrix [14], it is usual to set a gap penalty of -4.

The ILP model for multiple sequence alignment defines the same variables $x_{i,j,p,q}$ as the ILP model for longest common subsequence, along with all but the first constraint of the ILP model for longest common subsequence, and the goal of this model is to maximize the following scoring function,

$$\begin{aligned} \max \sum_{i,j,p,q} & m_{i,j,p,q} x_{i,j,p,q} \\ & + \text{gap} \sum_{i,j,p} (1 - \sum_{q} x_{i,j,p,q}) \\ & + \text{gap} \sum_{i,j,q} (1 - \sum_{p} x_{i,j,p,q}) \end{aligned}$$

where i ranges over $1,\ldots,n-1$, $j = i+1$, p ranges over $1,\ldots,|s_i|$, and q ranges over $1,\ldots,|s_j|$. The first term corresponds to matches and mismatches, the second term corresponds to gap penalties for insertions, and the third term corresponds to gap penalties for deletions.

2.2 Second ILP Model for Multiple Sequence Alignment

The huge number of constraints in the first ILP models for longest common subsequence and multiple sequence alignment, can be significatively reduced by encoding a multiple alignment using integer variables instead of just binary variables.

The new model for the longest common subsequence problem also takes takes the following input parameters: n, the number of sequences, and $S = \{s_1, \ldots, s_n\}$, a set of n sequences. Also, let us define again an additional parameter, $m_{i,j,p,q}$, for each pair of consecutive sequences s_i and s_j, position $p = 1, \ldots, |s_i|$, and position $q = 1, \ldots, |s_j|$, as follows:

$$m_{i,j,p,q} = \begin{cases} 1, & \text{if } s_i[p] = s_j[q] \\ 0, & \text{otherwise} \end{cases}$$

Let us now define an integer variable x_{ip} for each sequence s_i and position $p = 1, \ldots, |s_i|$, where $x_{i,p}$ is the position or column number of symbol $s_i[p]$ in a *longest common subsequence*. Thus, the new variables x also encode a solution to an LCS problem instance, as enforced by the following linear constraints:

1. The symbols that comprise a solution must appear in the same order in all the sequences. This means, there cannot be "crosses" between these symbols in any pair of consecutive sequences. For $i = 1, \ldots, n-1$ and $p_1, q_1 = 1, \ldots, |s_i| - 1$,

$$x_{i,p} \leqslant x_{i,p+1} - 1$$

2. Different symbols cannot be aligned at a same postion or column in a longest common subsequence. That is, $m_{i,j,p,q} = 0 \Rightarrow x_{i,p} \neq x_{j,q}$. This implication can be turned into linear constraints using the big-M method [5]. For $i = 1, \ldots, n-1$, $j = i+1$, $p = 1, \ldots, |s_i|$, $q = 1, \ldots, |s_j|$,

$$y_{1\,i,j,p,q} + y_{2\,i,j,p,q} = 1$$

$$x_{j,q} + 0.5 \leqslant x_{i,p} + M(1 - y_{1\,i,j,p,q})$$

$$x_{i,p} + 0.5 \leqslant x_{j,q} + M(1 - y_{2\,i,j,p,q})$$

where y_1 and y_2 are new binary variables, for each pair of consecutive sequences s_i and s_j, position p of sequence s_i, and position q of sequence s_j.

The goal of this new ILP model for the longest common subsequence problem is to minimize the sum of the positions,

$$\min \sum_{i,p} x_{i,p}$$

where i ranges over $1, \ldots, n$ and p ranges over $1, \ldots, |s_i|$.

Notice that this does not guarantee that the length of the common subsequence be maximized. Nevertheless, this ILP model can be easily extended to find an optimal solution to the multiple sequence alignment problem, as follows.

Let us redefine again the parameter m in terms of nucleotide or amino acid substitution matrices:

$$m_{i,j,p,q} = \text{substitution\_matrix}[s_i[p], s_j[q]]$$

The new ILP model for multiple sequence alignment defines the same variables $x_{i,p}$ as the new ILP model for longest common subsequence, along with the following linear constraints:

1. The symbols that comprise a solution must appear in the same order in all the sequences. This means, there cannot be "crosses" between these symbols in any pair of consecutive sequences. For $i = 1,\ldots,n-1$ and $p_1,q_1 = 1,\ldots,|s_i|-1$,

$$x_{i,p} \leqslant x_{i,p+1} - 1$$

2. Variable y encodes a multiple alignment. That is, $y_{i,j,p,q} = 1 \Leftrightarrow x_{i,p} = x_{j,q}$. This equivalence can also be turned into linear constraints using the big-M method [5]. For $i = 1,\ldots,n-1$, $j = i+1$, $p = 1,\ldots,|s_i|$, $q = 1,\ldots,|s_j|$,

$$x_{j,q} \leqslant x_{i,p} + M\,(1 - y_{i,j,p,q})$$

$$x_{i,p} \leqslant x_{j,q} + M\,(1 - y_{i,j,p,q})$$

where y is a new binary variable, for each pair of consecutive sequences s_i and s_j, position p of sequence s_i, and position q of sequence s_j.

The goal of this new ILP model for the multiple sequence alignment problem is to maximize the scoring function of the alignment,

$$\max \sum_{i,j,p,q} y_{i,j,p,q}\, m_{i,j,p,q}$$

where i ranges over $1,\ldots,n-1$, $j = i+1$, p ranges over $1,\ldots,|s_i|$, and q ranges over $1,\ldots,|s_j|$.

Notice that, unlike the first ILP model for multiple sequence alignment, this second ILP model does not take gap penalties into account.

3 Results and Discussion

We tested our implementation of the two ILP models for multiple sequence alignment, written in AMPL [8], with a dataset of 14 N-acetylglucosamine-binding protein sequences [3]. We solved these models using AMPL version 2018.10.22 and Gurobi Optimizer version 8.1.0, on a personal computer with an Intel Core i7-8550U quad-core processor at 1.80 GHz and 32 GB of memory running Ubuntu 18.04 LTS.

Some of the parameters that have a strong influence in the solver running time of an ILP model are: the number of variables, the number of constraints, and the number of non-zeros in the concrete ILP model being solved. In order to simplify the analysis, let us consider the MSA of n sequences of length at most ℓ.

The first ILP model for multiple sequence alignment defines $n\ell^2$ binary variables and $3n\ell + 2n\ell^2 + n\ell^4$ constraints. The second ILP model for multiple sequence alignment, on the other hand, defines $n\ell$ integer variables and $n\ell^2$ binary variables, but only $n\ell + 2n\ell^2$ constraints.

With such a large number of constraints, the first ILP model for multiple sequence alignment uses an amount of memory far beyond the possibilities of a personal computer. For the multiple alignment of 14 sequence fragments of length 12, for instance, it defines 294,840 constraints; for sequence fragments of length 24, it defines 4,662,000 constraints; and for sequence fragments of length 36, it defines 23,552,424 constraints. For the N-acetylglucosamine-binding protein sequences from [3], which are between 41 and 48 amino acids long, the first ILP model for multiple sequence alignment defines between 39,609,444 and 74,384,352 constraints.

The second ILP model for multiple sequence alignment, on the other hand, only defines 4,200 constraints for the multiple alignment

Figure 2: Longest common subsequence of N-acetylglucosamine-binding protein sequences.

of 14 sequence fragments of length 12; 16,464 constraints for sequence fragments of length 24; and 36,792 constraints for sequence fragments of length 36. For the N-acetylglucosamine-binding protein sequences from [3], the second ILP model for MSA only defines between 47,642 and 65,184 constraints.

The number of non-zeros in the second ILP model for multiple sequence alignment are as follows: 11,540 for 14 sequence fragments of length 12; 45,572 for sequence fragments of length 24, and 102,068 for sequence fragment of length 36. For the N-acetylglucosamine-binding protein sequences from [3], the second ILP model for multiple sequence alignment has 158,880 non-zeros.

Figure 3 shows a multiple alignment of the N-acetylglucosamine-binding protein sequences from [3]: the best feasible solution to the second ILP model for multiple sequence alignment found in one hour of solver user time, which is also the optimal solution. We have also performed a multiple alignment of these sequences using three of the most popular multiple sequence alignment software tools: Clustal Omega [13], MUSCLE [7], and T-Coffee [11], using default parameter values. The resulting alignments, shown in the Supplementary materials, share with the alignment obtained with the second ILP model, the fact that most of the eight sulfur-containing cysteines are correctly aligned (six with Clustal Omega along 50 columns with 71 indels, eight with MUSCLE along 50 columns with 71 indels, six with T-Coffee along 59 columns with 197 indels, and five with the second ILP model along 49 columns with 57 indels).

4 Conclusions

We have reviewed current ILP models for multiple sequence alignment and presented a new ILP for multiple sequence alignment, based on a new ILP model for the longest common subsequence problem, that defines the same order

```
AATAHAQRCGEQGSNMECPNNLCCSQYGYCGMGGD-YCGKGCQNGACYT
VAATNAQTCGKQNDGMICPHNLCCSQFGYCGLGRD-YCGTGCQSGACCS
VGLVSAQRCGSQGGGGTCPALWCCSIWGWCGD-SEPYCGRTCENK-CWS
AATAQAQRCGEQGSNMECPNNLCCSQYGYCGMGGD-YCGKGCQNGACWT
AATAQAQRCGEQGSNMECPNNLCCSQYGYCGMGGD-YCGKGCQNGACWT
------QRCGEQGSGMECPNNLCCSQYGYCGMGGD-YCGKGCQNGACWT
SETVKSQNCGC----A--P-NLCCSQFGYCGST-DAYCGTGCRSGPCRS
RGSAE-Q-CGRQAGDALCPGGLCCSSYGWCGTTVD-YCGIGCQS-QCDG
AGPAAAQNCGCQ------PN-FCCSKFGYCGTT-DAYCGDGCQSGPCRS
AGPAAAQNCGCQ------PN-VCCSKFGYCGTT-DEYCGDGCQSGPCRS
RG-S-AEQCGQQAGDALCPGGLCCSSYGWCGTTAD-YCGDGCQSQ-CDG
RG-S-AEQCGRQAGDALCPGGLCCSFYGWCGTTVD-YCGDGCQSQ-CDG
TGVAIAEQCGRQAGGKLCPNNLCCSQWGWCGS-TDEYCSPDHNCQSNCK
------EQCGRQAGGKLCPNNLCCSQYGWCGS-SDDYCSPSKNCQSNCK
```

Figure 3: Multiple sequence alignment of N-acetylglucosamine-binding protein sequences.

of magnitude of variables but two orders of magnitude fewer constraints. We believe that, with more research and improvements, the latter can be an interesting tool for finding optimal solutions to the multiple sequence alignment problem in many practical situations.

Further work includes extending the first ILP model for multiple sequence alignment in order to also take a gap extension penalty into account, extending the second ILP model for multiple sequence alignment in order to take both gap opening and gap extension penalties into account, and comparing the quality and running time of the ILP models for multiple sequence alignment with other multiple sequence alignment methods, using a benchmark alignment database for the evaluation of multiple alignments such as BAliBASE [15].

References

[1] E. Althaus and S. Canzar. A Lagrangian relaxation approach for the multiple sequence alignment problem. *Journal of Combinatorial Optimization*, 16(2):127–154, 2008.

[2] E. Althaus, A. Caprara, H.-P. Lenhof, and K. Reinert. Multiple sequence alignment with arbitrary gap costs: Computing an optimal solution using polyhedral combinatorics. *Bioinformatics*, 18(suppl_2):S4–S16, 2002.

[3] E. Althaus, A. Caprara, H.-P. Lenhof, and K. Reinert. A branch-and-cut algorithm for multiple sequence alignment. *Mathematical Programming*, 105(2-3):387–425, 2006.

[4] R. Beal, T. Afrin, A. Farheen, and D. Adjeroh. A new algorithm for the LCS problem with application in compressing

genome resequencing data. *BMC Genomics*, 17(4):544, 2016.

[5] P. Belotti, L. Liberti, A. Lodi, G. Nannicini, and A. Tramontani. Disjunctive inequalities: Applications and extensions. *Wiley Encyclopedia of Operations Research and Management Science*, 2010.

[6] C. Blum and M. J. Blesa. Hybrid techniques based on solving reduced problem instances for a longest common subsequence problem. *Applied Soft Computing*, 62:15–28, 2018.

[7] R. C. Edgar. MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Research*, 32(5):1792–1797, 2004.

[8] R. Fourer, D. M. Gay, and B. W. Kernighan. *AMPL: A Modeling Language for Mathematical Programming*. Cengage Learning, Boston, Massachussets, 2nd edition, 2002.

[9] D. Gusfield. *Integer Linear Programming in Computational and Systems Biology: An Entry-Level Text and Course*. Cambridge University Press, Cambridge, UK, 2019.

[10] J. Kececioglu. The maximum weight trace problem in multiple sequence alignment. In A. Apostolico, M. Crochemore, Z. Galil, and U. Manber, editors, *Proc. 4th Annual Symposium on Combinatorial Pattern Matching*, volume 684 of *Lecture Notes in Computer Science*, pages 106–119. Springer, 1993.

[11] C. Notredame, D. G. Higgins, and J. Heringa. T-Coffee: A novel method for fast and accurate multiple sequence alignment. *Journal of Molecular Biology*, 302(1):205–217, 2000.

[12] E. Pappalardo, P. M. Pardalos, and G. Stracquadanio. *Optimization Approaches for Solving String Selection Problems*. SpringerBriefs in Optimization. Springer, 2013.

[13] F. Sievers and D. G. Higgins. Clustal Omega. *Current Protocols in Bioinformatics*, 48(1):3–13, 2014.

[14] M. P. Styczynski, K. L. Jensen, I. Rigoutsos, and G. Stephanopoulos. BLOSUM62 miscalculations improve search performance. *Nature Biotechnology*, 26(3):274, 2008.

[15] J. D. Thompson, F. Plewniak, and O. Poch. BAliBASE: A benchmark alignment database for the evaluation of multiple alignment programs. *Bioinformatics*, 15(1):87–88, 1999.

[16] L. Wang and T. Jiang. On the complexity of multiple sequence alignment. *Journal of Computational Biology*, 1(4):337–348, 1994.

Supplementary material

Multiple Sequence Alignment with Clustal Omega

Multiple sequence alignment obtained with Clustal Omega [13] of the N-acetylglucosamine-binding protein sequences from [3].

```
AATAHAQRCGEQGSNMECPNNLCCSQYGYCGMGGDYCGK--GCQNGACYT
VAATNAQTCGKQNDGMICPHNLCCSQFGYCGLGRDYCGT--GCQSGACCS
VGLVSAQRCGSQGGGGTCPALWCCSIWGWCGDSEPYCGR--TCENKC-WS
AATAQAQRCGEQGSNMECPNNLCCSQYGYCGMGGDYCGK--GCQNGACWT
AATAQAQRCGEQGSNMECPNNLCCSQYGYCGMGGDYCGK--GCQNGACWT
------QRCGEQGSGMECPNNLCCSQYGYCGMGGDYCGK--GCQNGACWT
SETVKSQ------NCGCAPNLCCSQFGYCGSTDAYCGT--GCRSGPCRS
--RGSAEQCGRQAGDALCPGGLCCSSYGWCGTTVDYCGI--GCQSQCDG-
AGPAAAQ------NCGCQPNFCCSKFGYCGTTDAYCGD--GCQSGPCRS
AGPAAAQ------NCGCQPNVCCSKFGYCGTTDEYCGD--GCQSGPCRS
--RGSAEQCGQQAGDALCPGGLCCSSYGWCGTTADYCGD--GCQSQCDG-
--RGSAEQCGRQAGDALCPGGLCCSFYGWCGTTVDYCGD--GCQSQCDG-
TGVAIAEQCGRQAGGKLCPNNLCCSQWGWCGSTDEYCSPDHNCQSNCK--
------EQCGRQAGGKLCPNNLCCSQYGWCGSSDDYCSPSKNCQSNCK--
```

Multiple Sequence Alignment with MUSCLE

Multiple sequence alignment obtained with MUSCLE [7] of the N-acetylglucosamine-binding protein sequences from [3].

```
AATAHAQRCGEQGSNMECPNNLCCSQYGYCGMGGDYCGKG--CQNGACYT
VAATNAQTCGKQNDGMICPHNLCCSQFGYCGLGRDYCGTG--CQSGACCS
VGLVSAQRCGSQGGGGTCPALWCCSIWGWCGDSEPYCGRT--CEN-KCWS
AATAQAQRCGEQGSNMECPNNLCCSQYGYCGMGGDYCGKG--CQNGACWT
AATAQAQRCGEQGSNMECPNNLCCSQYGYCGMGGDYCGKG--CQNGACWT
------QRCGEQGSGMECPNNLCCSQYGYCGMGGDYCGKG--CQNGACWT
SETVKSQNCG-------CAPNLCCSQFGYCGSTDAYCGTG--CRSGPCRS
RGSA--EQCGRQAGDALCPGGLCCSSYGWCGTTVDYCGIG--CQS-QCDG
AGPAAAQNCG-------CQPNFCCSKFGYCGTTDAYCGDG--CQSGPCRS
AGPAAAQNCG-------CQPNVCCSKFGYCGTTDEYCGDG--CQSGPCRS
RGSA--EQCGQQAGDALCPGGLCCSSYGWCGTTADYCGDG--CQS-QCDG
RGSA--EQCGRQAGDALCPGGLCCSFYGWCGTTVDYCGDG--CQS-QCDG
TGVAIAEQCGRQAGGKLCPNNLCCSQWGWCGSTDEYCSPDHNCQS-NCK-
------EQCGRQAGGKLCPNNLCCSQYGWCGSSDDYCSPSKNCQS-NCK-
```

Multiple Sequence Alignment with T-Coffee

Multiple sequence alignment obtained with T-Coffee [11] of the N-acetylglucosamine-binding protein sequences from [3].

```
AATAHAQ------RCGE--QGS-NMECPNNLCCSQYGYCGMGGDYCGKG--CQNGACYT
------VAATNAQTCGK--QND-GMICPHNLCCSQFGYCGLGRDYCGTG--CQSGACCS
VGLVSAQ------RCGS--QGG-GGTCPALWCCSIWGWCGDSEPYCGRT--CENKCWS-
AATAQAQ------RCGE--QGS-NMECPNNLCCSQYGYCGMGGDYCGKG--CQNGACWT
AATAQAQ------RCGE--QGS-NMECPNNLCCSQYGYCGMGGDYCGKG--CQNGACWT
------Q------RCGE--QGS-GMECPNNLCCSQYGYCGMGGDYCGKG--CQNGACWT
--------------CSETVKSQ-NCGCAPNLCCSQFGYCGSTDAYCGTG--CRSGPCRS
--RGSAE------QCGR--QAG-DALCPGGLCCSSYGWCGTTVDYCGIG--CQSQCDG-
--------------CGP--AAAQNCGCQPNFCCSKFGYCGTTDAYCGDG--CQSGPCRS
--------------CGP--AAAQNCGCQPNVCCSKFGYCGTTDEYCGDG--CQSGPCRS
--RGSAE------QCGQ--QAG-DALCPGGLCCSSYGWCGTTADYCGDG--CQSQCDG-
--RGSAE------QCGR--QAG-DALCPGGLCCSFYGWCGTTVDYCGDG--CQSQCDG-
TGVAIAE------QCGR--QAG-GKLCPNNLCCSQWGWCGSTDEYCSPDHNCQSNCK--
------E------QCGR--QAG-GKLCPNNLCCSQYGWCGSSDDYCSPSKNCQSNCK--
```