Biodesign

pISSN 2288-6982 | eISSN 2288-7105
https://doi.org/10.34184/kssb.2025.13.1.1

PROTOCOL P 1-5

A rapid procedure to prepare for input file in a FASTA format for Alphafold3 local version

Minho Park[1], Ungyu Lee[1] and Nam-Chul Ha[1,2,*]

[1]Research Institute of Agriculture and Life Sciences, Department of Agricultural Biotechnology, CALS, Seoul National University, Seoul 08826, Republic of Korea

[2]Center for Food and Bioconvergence, Department of Agricultural Biotechnology, Interdisciplinary Programs in Agricultural Genomics, CALS, Seoul National University, Seoul 08826, Republic of Korea

*Correspondence: hanc210@snu.ac.kr

AlphaFold3 represents a significant advancement in protein structure prediction, extending its utility to complex biomolecular systems such as homomers, heteromers, nucleic acids, and ligand-bound structures. However, the local version of AlphaFold3 exclusively accepts JSON file inputs, which limits its practicality for simple structure predictions and high-throughput applications. This study developed a streamlined workflow utilizing FASTA format for input preparation, enhancing both the simplicity and efficiency of AlphaFold3 for local use, particularly in high-throughput structural predictions. Tools and scripts supporting this workflow are available at https://github.com/snufoodbiochem/Alphafold3_tools.

INTRODUCTION

AlphaFold3 (Abramson et al., 2024), the latest iteration of DeepMind's protein structure prediction framework, offers groundbreaking capabilities for modeling multi-component biomolecular systems and incorporating post-translational modifications (PTMs). However, the local version of AlphaFold3 relies on JSON file inputs, which can complicate workflows for simple structure predictions and high-throughput applications. In contrast, the server version provides a more user-friendly interface that supports simpler input formats, streamlining routine tasks and reducing technical barriers.

FASTA format, a widely used standard for representing nucleotide or protein sequences, is favored for its simplicity and compatibility with bioinformatics tools. Each sequence entry begins with a single-line description starting with a ">" symbol, followed by an identifier and optional metadata. The sequence itself is written on subsequent lines, making FASTA compact and easy to process for basic sequence storage and analysis. However, its simplicity limits its ability to incorporate complex annotations or metadata necessary for advanced prediction tools like AlphaFold3.

JSON format, on the other hand, is a structured and flexible format commonly used for data exchange. It supports hierarchical data representation, enabling the storage of sequences alongside detailed metadata such as annotations, structural constraints, and additional parameters. While JSON is ideal for integrating complex data in modern bioinformatics workflows, its format is less intuitive for direct manual input compared to FASTA.

To bridge this gap, a hybrid approach is proposed: leveraging FASTA for initial sequence input and converting it into a JSON structure to add or modify more detailed metadata as needed. This method combines the simplicity of FASTA with the advanced capabilities of JSON, making it particularly useful for enhancing the usability of AlphaFold3 local version. By streamlining the process of input preparation, this approach facilitates efficient sequence prediction and enables the incorporation of advanced metadata, such as PTMs or custom constraints, into the workflow.

This study outlines an input preparation strategy that utilizes FASTA as a base format, enabling the efficient use of AlphaFold3's local version for both individual predictions and large-scale applications.

RESULTS

Development of code for converting FASTA to JSON format for AlphaFold3 local version

To develop the FASTA-to-JSON conversion code for AlphaFold3, we referred extensively to the official AlphaFold3 input documentation (https://github.com/google-deepmind/alphafold3/blob/main/docs/input.md). We identified that most of AlphaFold3's functionalities could be seamlessly adapted to work with FASTA format by introducing modifications in the label line.

Using ChatGPT (OpenAI, 2022), we initially created a draft version of the Python code, gradually enhancing it by incorporating additional functions step by step. Throughout this iterative process, we identified and resolved bugs, improving the code's reliability. Although we do not specialize in Python programming, ChatGPT proved invaluable in helping us develop the tool tailored to convert FASTA format inputs into JSON format compatible with AlphaFold3 local version. The following sections outline the usage of the Python code we developed in detail.

(A) **complex.fasta**

```
>protein1 #2
MNIRDLEYLVALAEHRHFRRAADSCHVSQPTLSGQIRKLEDELGVMLLERTSRKVLFTQAGMLLVDQARTVLREVKVLKEMASQQ
GETMSGPLHIGLIPTVGPYLLPHIIPMLHQTFPKLEMYLHEAQTHQLLAQLDSGKLDCVILALVKESEAFIEVPLFDEPMLLAIY
EDHPWANRECVPMADLAGEKLLMLEDGHCLRDQAMGFCFEAGADEDTHFRATSLETLRNMVAAGSGITLLPALAVPPERKRDGVV
YLPCIKPEPRRTIGLVYRPGSPLRSRYEQLAEAIRARMDGHFDKVLKQAV

>protein2
MFQQEVTITAPNGLHTRPAAQFVKEAKGFTSEITVTSNGKSASAKSLFKLQTLGLTQGTVVTISAEGEDEQKAVEHLVKLMAELE
```

`python3 fasta2json.py complex.fasta`

(B) **complex.json**

```
{
  "name": "complex",
  "modelSeeds": [ 1],
  "sequences": [
    {"protein": {
        "id": [ "A", "B" ],
        "sequence":
"MNIRDLEYLVALAEHRHFRRAADSCHVSQPTLSGQIRKLEDELGVMLLERTSRKVLFTQAGMLLVDQARTVLREVKVLKEMASQ
QGETMSGPLHIGLIPTVGPYLLPHIIPMLHQTFPKLEMYLHEAQTHQLLAQLDSGKLDCVILALVKESEAFIEVPLFDEPMLLAI
YEDHPWANRECVPMADLAGEKLLMLEDGHCLRDQAMGFCFEAGADEDTHFRATSLETLRNMVAAGSGITLLPALAVPPERKRDGV
VYLPCIKPEPRRTIGLVYRPGSPLRSRYEQLAEAIRARMDGHFDKVLKQAV",
        "modifications": []}
    },
    {"protein": {
        "id": [ "C" ],
        "sequence":
"MFQQEVTITAPNGLHTRPAAQFVKEAKGFTSEITVTSNGKSASAKSLFKLQTLGLTQGTVVTISAEGEDEQKAVEHLVKLMAEL
E",
        "modifications": []}
    }],
  "bondedAtomPairs": [],
  "dialect": "alphafold3",
  "version": 1
}
```

Alphafold3

(C)

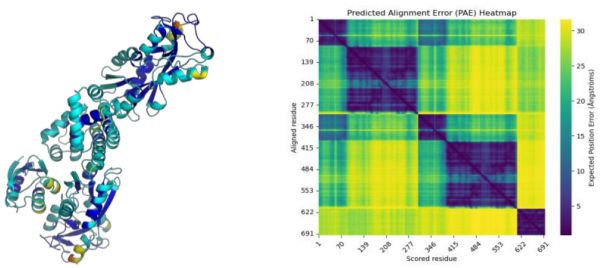

FIGURE 1 I Example of FASTA and JSON formats in protein structure prediction procedure using Alphafold3 local version. (A) An example FASTA file containing the protein sequences of protein 1 and protein 2. (B) An example of JSON file generated by converting the complex.fasta using the fasta2json.py script. The JSON format organizes the protein sequence data for use in structural prediction tools. (C) The output generated by Alphafold3, including the predicted 3D structure of the proteins and predicted alignment error (PAE) heatmap for assessing the accuracy of the model.

A single protein sequence

Users can prepare a FASTA file with a label line beginning with '>', followed by the protein sequence in single-letter amino acid code. JSON files for AlphaFold3 local version do not permit lowercase letters, spaces, or newlines in the sequence. However, the provided Python script automatically converts lowercase letters to uppercase and removes spaces and newlines, allowing users to include these elements when preparing the FASTA file. It is important to keep the label line concise, as it must adhere to the new rules established for JSON file flexibility.

Example: an OxyR monomer

```
>oxyR

MNIRDLEYLVALAEHRHFRRAADSCHVSQPTLSGQIRKLEDELGVM
LLERTSRKVLFTQAGMLLVDQARTVLREVKVLKEMASQQGETMSGP
LHIGLIPTVGPYLLPHIIPMLHQTFPKLEMYLHEAQTHQLLAQLDS
GKLDCVILALVKESEAFIEVPLFDEPMLLAIYEDHPWANRECVPMA
DLAGEKLLMLEDGHCLRDQAMGFCFEAGADEDTHFRATSLETLRNM
VAAGSGITLLPALAVPPERKRDGVVYLPCIKPEPRRTIGLVYRPGS
PLRSRYEQLAEAIRARMDGHFDKVLKQAV
```

Homomers

Homomers are annotated by appending #<oligomer number> to the sequence label.

Example: an OxyR dimer

```
>oxyR #2

MNIRDLEYLVALAEHRHFRRAADSCHVSQPTLSGQIRKLEDELGVM
LLERTSRKVLFTQAGMLLVDQARTVLREVKVLKEMASQQGETMSGP
LHIGLIPTVGPYLLPHIIPMLHQTFPKLEMYLHEAQTHQLLAQLDS
GKLDCVILALVKESEAFIEVPLFDEPMLLAIYEDHPWANRECVPMA
DLAGEKLLMLEDGHCLRDQAMGFCFEAGADEDTHFRATSLETLRNM
VAAGSGITLLPALAVPPERKRDGVVYLPCIKPEPRRTIGLVYRPGS
PLRSRYEQLAEAIRARMDGHFDKVLKQAV
```

Heteromers

Additional sequences for heteromeric systems are provided in sequence.

Example: a heteromer consisting of an OxyR dimer and an Hpr monomer

```
>oxyR #2

MNIRDLEYLVALAEHRHFRRAADSCHVSQPTLSGQIRKLEDELGVM
LLERTSRKVLFTQAGMLLVDQARTVLREVKVLKEMASQQGETMSGP
LHIGLIPTVGPYLLPHIIPMLHQTFPKLEMYLHEAQTHQLLAQLDS
GKLDCVILALVKESEAFIEVPLFDEPMLLAIYEDHPWANRECVPMA
DLAGEKLLMLEDGHCLRDQAMGFCFEAGADEDTHFRATSLETLRNM
VAAGSGITLLPALAVPPERKRDGVVYLPCIKPEPRRTIGLVYRPGS
PLRSRYEQLAEAIRARMDGHFDKVLKQAV

>hpr

MFQQEVTITAPNGLHTRPAAQFVKEAKGFTSEITVTSNGKSASAKS
LFKLQTLGLTQGTVVTISAEGEDEQKAVEHLVKLMAELE
```

DNA and RNA

FASTA format also supports nucleic acid sequences, labeled as dna for DNA or rna for RNA.

The python code recognizes dna and rna keywords in the label line. The code also supports the multimer option of #<oligomer number> like the protein labels.

Example: two DNA molecules

```
>dna#2

GATACAGACCATTTT
```

Example: an RNA molecule

```
>rna

GAUACAGACCAUUU
```

Ligands

AlphaFold3 local version introduces enhanced support for modeling protein-ligand complexes, accommodating a wide range of ligands and ions to improve structural predictions involving bound cofactors or substrates (Abramson et al., 2024). Ligands are classified into two categories based on their representation format: those with a Chemical Component Dictionary (CCD) code and those described using the SMILES (Simplified Molecular Input Line Entry System) notation. This flexibility enables users to include ligands ranging from small molecules like ATP and ADP to metal ions or custom ligands.

For ligands with a CCD code, users should specify the ligand using its 3-letter code. Ions, which are treated as ligands, can include MG (magnesium), ZN (zinc), CL (chloride), CA (calcium), NA (sodium), MN (manganese), K (potassium), FE (iron), CU (copper), and CO (cobalt), as listed in the AlphaFold3 documentation (https://github.com/google-deepmind/alphafold/blob/main/server/README.md). These ligands can be directly input with the >ligand label, simplifying the inclusion of standard cofactors.

```
>ligand

ATP

>ligand

MG
```

For ligands not associated with a CCD code, the SMILES format is used. Users can define these ligands under the >smile label. This method allows the representation of novel or customized ligands without requiring CCD registration. For example:

```
>smile

CC(=O)OC1C[NH+]2CCC1CC21
```

This flexibility in ligand representation extends the capabilities of AlphaFold3 local version, enabling high-accuracy modeling of complex systems that include both standard and non-standard ligands. A comprehensive list of CCD codes is available via the EBI Ligand Search tool (Golovin et al., 2004; Dimitropoulos et al., 2006), providing a valuable resource for users to identify and incorporate standard ligands into their predictions.

Post-translational modifications (PTMs)

AlphaFold3 local version supports the inclusion of post-translational modifications (PTMs) for proteins, as well as modifications on DNA and RNA molecules, as detailed on the official AlphaFold3 website (Abramson et al., 2024). PTMs are annotated using Chemical Component Dictionary (CCD) codes, similar to ligands, and are added to the label with the format &<amino acid number>_<PTM type>. This allows users to specify PTMs directly in the input sequence.

Example: If Cys199 in an OxyR dimer are modified to cysteine sulfenic acid:

```
>oxyR &199_CSO #2

MNIRDLEYLVALAEHRHFRRAADSCHVSQPTLSGQIRKLEDELGVM
LLERTSRKVLFTQAGMLLVDQARTVLREVKVLKEMASQQGETMSGP
LHIGLIPTVGPYLLPHIIPMLHQTFPKLEMYLHEAQTHQLLAQLDS
GKLDCVILALVKESEAFIEVPLFDEPMLLAIYEDHPWANRECVPMA
DLAGEKLLMLEDGHCLRDQAMGFCFEAGADEDTHFRATSLETLRNM
VAAGSGITLLPALAVPPERKRDGVVYLPCIKPEPRRTIGLVYRPGS
PLRSRYEQLAEAIRARMDGHFDKVLKQAV
```

The working PTMs we tested are listed in Table 1 below. For a comprehensive list of PTMs, refer to the EBI Ligand Search (Golovin et al., 2004; Dimitropoulos et al., 2006). PTMs for DNA and RNA were also tested and listed in https://github.com/google-deepmind/alphafold/blob/main/server/README.md (Jumper et al., 2021), with information sourced from the official AlphaFold3 input examples.

Ligands composed of multiple molecules: handling glycan chains

For ligands that consist of multiple molecules, such as glycan chains, the individual molecules or residues should be listed in the sequence section, separated by commas. Additionally, the bonds between residues must be defined in the label line using the format &<residue number>_<atom name>_<residue number>_<atom name>.

For example, if N-acetylglucosamine (NAG) and fucose (FUC) are linked via a 1,4-glycosidic bond, the ligand is annotated as:

TABLE 1 I Common post-translational modifications (PTMs) supported by AlphaFold3

PTM code	Description	Amino acid
3HY	3-hydroxylated proline	Proline
P1L	S-palmitoyl-L-cysteine	Cysteine
PTR	Phosphotyrosine	Tyrosine
NMM	Methylarginine	Arginine
LYZ	Hydroxylysine	Lysine
CSO	Cysteine sulfenic acid	Cysteine
CSD	Cysteine sulfinic acid	Cysteine
OCS	Cysteine sulfonic acid	Cysteine
MYK	Myristoyllysine	Lysine
SEP	Phosphorylated serine	Serine
TPO	Phosphorylated threonine	Threonine
NEP	N1-phosphonohistidine	Histidine
HIP	ND1-phosphonohistidine	Histidine
ALY	N(6)-acetyllysine	Lysine
MLY	N-dimethyl-lysine	Lysine
M3L	N-trimethyllysine	Lysine
MLZ	N-methyl-lysine	Lysine
2MR	N3, N4-dimethylarginine	Arginine
AGM	5-methyl-arginine	Arginine
MCS	Malonyl cysteine	Cysteine
HYP	4-hydroxyproline	Proline
AHB	Beta-hydroxyasparagine	Asparagine
SNN	L-3-aminosuccinimide	Asparagine
SNC	S-nitroso-cysteine	Cysteine
TRF	N1-formyl-tryptophan	Tryptophan
KCR	N-6-crotonyl-L-lysine	Lysine
CIR	Citrulline	Arginine
YHA	Homocitrulline	Lysine

This table's "Code" and "Description" columns were extracted from the PDBeChem database (Golovin et al., 2004; Dimitropoulos et al., 2006).

```
>ligand &1_O4_2_C1

NAG,FUC
```

This approach ensures precise representation of multi-molecular ligands and their connectivity, enabling accurate structural modeling in AlphaFold3.

This structured approach to annotating PTMs enhances AlphaFold3's capacity to model complex biological systems, facilitating accurate structural predictions that incorporate specific chemical modifications.

Features not supported by this python script

The local version of AlphaFold3 also supports advanced features such as multiple sequence alignments (MSAs) for RNA and proteins, MSA pairing, and structural templates, all of which require input in JSON format. Additionally, specific ligands

can be included using the CCD mmCIF format. However, implementing these functionalities directly from the FASTA format is not straightforward. We recommend converting the output JSON file generated from the FASTA input and manually adding or editing the required details to utilize these advanced features effectively.

Conversion to JSON format

For scenarios requiring JSON inputs, the following conversion command can be used:

```
python3 fasta2json.py <input fasta>
```

DISCUSSION

This study highlights the potential of using FASTA format as a streamlined alternative for input preparation in the AlphaFold3 local version, addressing some of the challenges posed by the default reliance on JSON files. The use of FASTA format offers distinct advantages, particularly for simplifying input in simple structure predictions and enabling high-throughput workflows. These benefits make FASTA an attractive option for routine or large-scale protein structure predictions, especially when compared to the more complex and labor-intensive JSON-based input preparation.

The efficiency of FASTA in batch processing aligns with the scalability demands of high-throughput applications, bridging a critical gap in the AlphaFold3 local version. This approach also provides a practical complement to the AlphaFold3 server, which, while user-friendly, is limited in its capacity for handling large datasets. The combination of FASTA and the AlphaFold3 local version thus presents a robust solution for researchers balancing simplicity and scalability.

However, the intrinsic limitations of the FASTA format must be acknowledged. FASTA cannot natively support the complex metadata required by AlphaFold3 local version for advanced features such as post-translational modifications (PTMs), ligands, and atom-specific annotations. While the FASTA-to-JSON converter mitigates some of these challenges, it cannot fully replicate the flexibility and precision offered by JSON. For example, handling hybrid structures, multi-chain assemblies, or intricate functional complexes still necessitates the detailed specification afforded by JSON input.

Moreover, the dual support for both FASTA and JSON formats may introduce potential confusion and increase the risk of bugs, particularly in workflows that involve transitioning between the two formats. These risks underline the importance of positioning FASTA-based workflows as a supplementary tool, primarily for straightforward predictions or high-throughput use cases, rather than as a replacement for JSON.

In conclusion, while the FASTA format enhances accessibility and efficiency for certain applications in AlphaFold3 local version, its utility remains context-dependent. FASTA workflows are best suited for simple protein predictions and large-scale datasets, whereas JSON should remain the primary standard for complex and detailed structural modeling. Balancing these complementary approaches can maximize the flexibility and utility of AlphaFold3 for a wide range of research applications. Tools and resources for implementing this approach are available at https://github.com/snufoodbiochem/Alphafold3_tools.

ACKNOWLEDGEMENTS

This research was supported by the Bio & Medical Technology Development Program of the National Research Foundation (NRF) funded by the Ministry of Science & ICT (RS-2024-00344154 to N.-C.H.).

CONFLICT OF INTEREST

The authors declare no potential conflicts of interest.

Original Submission: Dec 4, 2024
Revised Version Received: Dec 24, 2024
Accepted: Dec 24, 2024

REFERENCES

Abramson, J., Adler, J., Dunger, J., Evans, R., Green, T., Pritzel, A., Ronneberger, O., Willmore, L., Ballard, A.J., Bambrick, J., Bodenstein, S.W., Evans, D.A., Hung, C.C., O'Neill, M., Reiman, D., et al. (2024). Accurate structure prediction of biomolecular interactions with AlphaFold 3. *Nature* **630**, 493-500.

Dimitropoulos, D., Ionides, J., and Henrick, K. (2006). Using MSDchem to search the PDB ligand dictionary. *Curr Protoc Bioinformatics* **14**, 14.3.1-14.3.21.

Golovin, A., Oldfield, T.J., Tate, J.G., Velankar, S., Barton, G.J., Boutselakis, H., Dimitropoulos, D., Fillon, J., Hussain, A., Ionides, J.M., John, M., Keller, P.A., Krissinel, E., McNeil, P., Naim, A., et al. (2004). E-MSD: an integrated data resource for bioinformatics. *Nucleic Acids Res* **32**, D211-D216.

Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., Bridgland, A., Meyer, C., Kohl, S.A.A., Ballard, A.J., Cowie, A., et al. (2021). Highly accurate protein structure prediction with AlphaFold. *Nature* **596**, 583-589.

OpenAI. (2022). ChatGPT [Internet]. OpenAI; [cited 2024 Mon day]. Available from: https://chatgpt.com/?model=gpt-4o.