nature portfolio

https://doi.org/10.1038/s41596-022-00763-x

Supplementary information

Uncovering structural ensembles from single-particle cryo-EM data using cryoDRGN

In the format provided by the authors and unedited

Supplementary protocol 1. Installing cryoDRGN version 0.3.5

1. Instructions for installing the latest version of cryoDRGN are available at https://github.com/zhonge/cryodrgn. For consistency with our results, we recommend using version 0.3.5, which we employed in this protocol. It can be installed using git as described below. To set up the conda environment, run the following commands:

```
conda create --name cryodrgn python=3.7
conda activate cryodrgn
conda install pytorch cudatoolkit=10.2 -c pytorch
conda install pandas seaborn scikit-learn
conda install umap-learn jupyterlab ipywidgets cufflinks-py
"nodejs>=15.12.0" -c conda-forge
conda update typing_extensions -c conda-forge
jupyter labextension install @jupyter-widgets/jupyterlab-manager
--no-build
jupyter labextension install jupyterlab-plotly --no-build
jupyter labextension install plotlywidget --no-build
jupyter lab build
```

   Critical step: Ensure that you install cudatoolkit and pytorch versions compatible with your graphics card and drivers. For example, your CUDA version is returned by the command `nvidia-smi`, and generally the latest pytorch version (built for your CUDA version and python 3.7) will be appropriate. See pytorch.org for more details on how to install pytorch.

2. Optionally install NVIDIA's Apex library to enable `--amp` acceleration via the following commands:

```
git clone https://github.com/NVIDIA/apex
cd apex
pip install -v --disable-pip-version-check --no-cache-dir ./
```

3. Optionally install the CUDA machine learning library for faster UMAP embeddings in analyze_convergence.py.

```
conda install cuml -c rapidsai-nightly -c rapidsai -c nvidia -c
conda-forge
```

4. Clone version 0.3.5 from GitHub:

```
git clone https://github.com/zhonge/cryodrgn.git
cd cryodrgn
git checkout tags/0.3.5
python setup.py install
```

Supplementary protocol 2. Creating a consensus refinement in cryoSPARC

1. Run an import particle stack job by specifying `L17Combine_weight_local.mrcs` as the particle data path and `Parameters.star` as the particle meta path. Note that the data sign needs to be flipped to dark-on-light.

2. Run an *ab initio* reconstruction job with default parameters.

3. Run a homogeneous refinement job with default parameters. Note that we generally suggest performing reconstructions without imposed symmetry (*i.e.* C1) as this preserves potentially interesting heterogeneity.

4. Copy the refined `particles.cs` file, whose name should resemble `cryosparc_P4_J33_004_particles.cs`, to the working cryoDRGN directory where the full dataset is stored.

Supplementary protocol 3. Setting up port forwarding via SSH

1. SSH port forwarding can be set up at the time of login using the following command and replacing `remote_username` and `remote_host_name` with the appropriate values:

   ```
   ssh -N -f -L localhost:8888:localhost:8888
   remote_username@remote_host_name
   ```

   If you are running your jupyter notebook on a worker node in a compute cluster, as opposed to a local workstation, we suggest the following alternative port forwarding command:

   ```
   ssh -t -t username@cluster-head-node -L 8888:localhost:8888 ssh
   active-worker-node -L 8888:localhost:8888
   ```

2. To open Jupyter notebook, enter the command `jupyter lab --no-browser --port 8888` into the terminal, and navigate to localhost:8888 in a web browser on your local computer.

Supplementary protocol 4. Generating segmented PDB chains for subunit occupancy analysis

1. Open PyMOL and use the command-line interface to retrieve an atomic model of the 70S ribosome from the PDB: `fetch 4ybb`

2. Delete atoms outside the region of interest. For example, to generate the segmented .pdb of the 5S rRNA, we use:

   ```
   sele not_5s, not chain CB
   ```

   Then select 'Remove atoms' from the drop-down 'Action' menu in the not_5s selection. This will delete all non-5S atoms.

3. Segment the map into chains if necessary. If you want to do occupancy analysis on whole protein subunits, this is likely unnecessary, as the chains are likely already defined in the atomic model. If you want to define your own subunits for occupancy analysis as we do here, you can do so using the `alter` command as shown below, again for the examples of the 5S rRNA:

   ```
   alter (resi 1-14,108-120), chain='A'
   alter (resi 15-27,60-68), chain='B'
   alter (resi 28-59), chain='C'
   alter (resi 78-99), chain='D'
   alter (resi 69-77,100-107), chain='E'
   ```

4. After you have made all the chain alterations, save the .pdb file with a new name, e.g. RNA_5S.pdb using the "Export Molecule" command. Note that to create more than 26 chains, you will need to use multiple .pdb files, each containing at most 26 chain IDs.

Supplementary protocol 5. Aligning segmented PDB models for subunit occupancy analysis

1. The .pdb files must now be aligned to your cryoDRGN sampled maps. Open one of the generated 500 maps (e.g. `vol_000.mrc`) in ChimeraX. Aim to select a map that has high occupancy of most elements of your structure to ensure a good alignment. Because maps with adjacent indices (e.g. vol_000 and vol_001) are often structurally similar as they are sampled from proximal locations in latent space, users are advised to find a mature map by downloading 20 random volumes from the set of 500.

2. Open all the .pdb files (`prots1.pdb`, `prots2.pdb`, `RNA_5S.pdb`, `RNA1.pdb`, `RNA2.pdb`, `RNA3.pdb`, `RNA4.pdb`). These should now be models #2-8 in your ChimeraX session.

3. Select models #2-8 with the command `select #2-8`. Provide a rough manual alignment between the selected atoms and the example map, using the 'Rotate model' and 'Move model' right mouse modes.

4. Having provided a rough manual alignment, use the Tools > Volume Data > Fit in Map option to fit your .pdb files in the map. Choose to fit 'selected atoms' in your example map, making sure that all the .pdb model files are still selected.

5. Save each of the .pdb files individually using File > Save, and selecting .pdb file type. Be sure you have the correct .pdb model selected in the Models selection box, and that you select the option to 'Save relative to model:', with the example map selected as the model.

Supplementary protocol 6. Identifying centroid volumes for subunit occupancy volume classes

1. Use pandas to load the dataframe you saved with information about which *k*-means 500 class each particle corresponds to.

```
df = pd.read_csv('kmeans500_df.csv', index_col = 0)
```

2. To save the volume classes defined by clustering in the `occupancy_analysis.ipynb` Jupyter notebook, run the cells in the 'Extract classes from clustering' section. This will save the class assignments as a .pkl file that you can load into the `cryoDRGN_viz.ipynb` notebook.

3. Open the volume class assignments .pkl file in the `cryoDRGN_viz.ipynb` notebook, changing the name or relative path of the .pkl file name as necessary in the code below.

```
classes = utils.load_pkl('../../vol_class.pkl')
```

4. Identify the nearest on-data point to the median z-coordinates of each class. The resulting variable `nearest_inds` contains the indices in your dataframe of the centroid particles. You can then generate volumes at these indices as before using the volume generation cells of the Jupyter notebook.

```
median_coords = np.empty([len(classes.keys()), z.shape[1]], dtype
= 'float64')
z_list = df.columns[df.columns.str.contains('z')]

for i in classes.keys():
    df.loc[df[df['Kmeans500'].isin(classes[i])].index,
'volume_class'] = i
    sub = df[df['volume_class'] == i][z_list]
    median_coords[i, :] = np.array(sub.median(axis = 0))

df_z = df[z_list]

neighbor_dists = pd.DataFrame(distance.cdist(median_coords, df_z,
'euclidean'))
nearest_inds = neighbor_dists.idxmin(axis = 1)
```

Supplementary Table 1: Residue and chain assignment for subunit occupancy analysis.

Subunit	PDB ID	PDB Chain	Residues	Segmented file name	Segmented file chain
H1	4YBB	CA	1-12,2895-2904	RNA1.pdb	A
H2	4YBB	CA	13-30,510-531	RNA1.pdb	B
H3	4YBB	CA	31-32,473-474	RNA1.pdb	C
H4	4YBB	CA	33-47,431-451	RNA1.pdb	D
H5	4YBB	CA	48-56,114-120	RNA1.pdb	E
H6	4YBB	CA	57-74	RNA1.pdb	F
H7	4YBB	CA	75-113	RNA1.pdb	G
H8	4YBB	CA	121-130	RNA1.pdb	H
H9	4YBB	CA	131-148	RNA1.pdb	I
H10	4YBB	CA	147-177	RNA1.pdb	J
H11	4YBB	CA	178-218,319-323	RNA1.pdb	K
H12	4YBB	CA	219-232	RNA1.pdb	L
H13	4YBB	CA	233-262	RNA1.pdb	M
H14	4YBB	CA	263-269,424-430	RNA1.pdb	N
H16	4YBB	CA	269-280,360-370	RNA1.pdb	O
H18	4YBB	CA	281-298,340-359	RNA1.pdb	P
H19	4YBB	CA	299-318	RNA1.pdb	Q
H20	4YBB	CA	324-339	RNA1.pdb	R
H21	4YBB	CA	371-404	RNA1.pdb	S
H22	4YBB	CA	405-423	RNA1.pdb	T
H23	4YBB	CA	452-472	RNA1.pdb	U
H24	4YBB	CA	475-509	RNA1.pdb	V
H25	4YBB	CA	532-561	RNA1.pdb	W
H25a	4YBB	CA	562-578	RNA1.pdb	X
H26	4YBB	CA	579-586,1251-1261	RNA1.pdb	Y
H27	4YBB	CA	587-602,655-670	RNA1.pdb	Z
H28	4YBB	CA	603-625	RNA2.pdb	A
H29	4YBB	CA	626-636	RNA2.pdb	B
H31	4YBB	CA	637-654	RNA2.pdb	C
H32	4YBB	CA	671-683,790-809	RNA2.pdb	D
H33	4YBB	CA	684-698,763-775	RNA2.pdb	E
H34	4YBB	CA	699-733	RNA2.pdb	F
H35	4YBB	CA	734-762	RNA2.pdb	G
H35a	4YBB	CA	776-789	RNA2.pdb	H
H36	4YBB	CA	810-821,1186-1195	RNA2.pdb	I
H37	4YBB	CA	822-835	RNA2.pdb	J
H38	4YBB	CA	836-942	RNA2.pdb	K
H39	4YBB	CA	943-973	RNA2.pdb	L
H40	4YBB	CA	974-990	RNA2.pdb	M
H41	4YBB	CA	991-1025,1133-1163	RNA2.pdb	N
H42	4YBB	CA	1026-1056,1103-1132	RNA2.pdb	O

H43	4YBB	CA	1057-1081	RNA2.pdb	P
H44	4YBB	CA	1087-1102	RNA2.pdb	Q
H45	4YBB	CA	1164-1185	RNA2.pdb	R
H46	4YBB	CA	1196-1250	RNA2.pdb	S
H26a	4YBB	CA	1262-1270,2010-2017	RNA2.pdb	T
H47	4YBB	CA	1271-1294	RNA2.pdb	U
H48	4YBB	CA	1295-1302,1640-1647	RNA2.pdb	V
H49	4YBB	CA	1303-1306,1622-1625	RNA2.pdb	W
H49b	4YBB	CA	1307-1313,1603-1608	RNA2.pdb	X
H50	4YBB	CA	1314-1338	RNA2.pdb	Y
H51	4YBB	CA	1339-1347,1599-1602	RNA2.pdb	Z
H52	4YBB	CA	1348-1382	RNA3.pdb	A
H53	4YBB	CA	1383-1404	RNA3.pdb	B
H54	4YBB	CA	1405-1417,1581-1598	RNA3.pdb	C
H55	4YBB	CA	1418-1428,1569-1580	RNA3.pdb	D
H49a	4YBB	CA	1609-1621	RNA3.pdb	E
H56	4YBB	CA	1429-1444,1547-1564	RNA3.pdb	F
H57	4YBB	CA	1445-1466	RNA3.pdb	G
H58	4YBB	CA	1467-1525	RNA3.pdb	H
H59	4YBB	CA	1526-1546	RNA3.pdb	I
H60	4YBB	CA	1626-1639	RNA3.pdb	J
H61	4YBB	CA	1648-1678,1990-2009	RNA3.pdb	K
H62	4YBB	CA	1679-1706	RNA3.pdb	L
H63	4YBB	CA	1707-1751	RNA3.pdb	M
H64	4YBB	CA	1758-1773,1977-1989	RNA3.pdb	N
H65	4YBB	CA	1774-1790	RNA3.pdb	O
H66	4YBB	CA	1791-1828	RNA3.pdb	P
H67	4YBB	CA	1829-1834,1970-1976	RNA3.pdb	Q
H68	4YBB	CA	1835-1905	RNA3.pdb	R
H69	4YBB	CA	1906-1924	RNA3.pdb	S
H71	4YBB	CA	1932-1969	RNA3.pdb	T
H72	4YBB	CA	2018-2042	RNA3.pdb	U
H73	4YBB	CA	2043-2057,2611-2625	RNA3.pdb	V
H74	4YBB	CA	2058-2074,2430-2451	RNA3.pdb	W
H75	4YBB	CA	2075-2092,2226-2245	RNA3.pdb	X
H76	4YBB	CA	2093-2114,2179-2196	RNA3.pdb	Y
H77	4YBB	CA	2115-2126,2169-2178	RNA3.pdb	Z
H78	4YBB	CA	2127-2168	RNA4.pdb	A
H79	4YBB	CA	2197-2225	RNA4.pdb	B
H80	4YBB	CA	2246-2258	RNA4.pdb	C
H81	4YBB	CA	2259-2281	RNA4.pdb	D
H82	4YBB	CA	2282-2286,2382-2390	RNA4.pdb	E
H83	4YBB	CA	2287-2296,2335-2344	RNA4.pdb	F
H84	4YBB	CA	2297-2321	RNA4.pdb	G

H85	4YBB	CA	2322-2334	RNA4.pdb	H
H86	4YBB	CA	2345-2371	RNA4.pdb	I
H87	4YBB	CA	2372-2381	RNA4.pdb	J
H88	4YBB	CA	2391-2429	RNA4.pdb	K
H89	4YBB	CA	2452-2504	RNA4.pdb	L
H90	4YBB	CA	2505-2517,2567-2586	RNA4.pdb	M
H91	4YBB	CA	2518-2546	RNA4.pdb	N
H92	4YBB	CA	2547-2561	RNA4.pdb	O
H93	4YBB	CA	2587-2610	RNA4.pdb	P
H94	4YBB	CA	2626-2643,2771-2788	RNA4.pdb	Q
H95	4YBB	CA	2644-2675	RNA4.pdb	R
H96	4YBB	CA	2676-2731	RNA4.pdb	S
H97	4YBB	CA	2732-2770	RNA4.pdb	T
H98	4YBB	CA	2789-2805	RNA4.pdb	U
H99	4YBB	CA	2806-2814,2886-2894	RNA4.pdb	V
H100	4YBB	CA	2815-2831	RNA4.pdb	W
H101	4YBB	CA	2832-2885	RNA4.pdb	X
H1_5S	4YBB	CB	1-14,108-120	RNA_5S.pdb	A
H2_5S	4YBB	CB	15-27,60-68	RNA_5S.pdb	B
H3_5S	4YBB	CB	28-59	RNA_5S.pdb	C
H4_5S	4YBB	CB	78-99	RNA_5S.pdb	D
H5_5S	4YBB	CB	69-77,100-107	RNA_5S.pdb	E
uL2	4YBB	CC	all	prots1.pdb	A
uL3	4YBB	CD	all	prots1.pdb	B
uL4	4YBB	CE	all	prots1.pdb	C
uL5	4YBB	CF	all	prots1.pdb	D
uL6	4YBB	CG	all	prots1.pdb	E
bL9	4YBB	CH	all	prots1.pdb	F
uL11	4YBB	CJ	all	prots1.pdb	G
uL13	4YBB	CK	all	prots1.pdb	H
uL14	4YBB	CL	all	prots1.pdb	I
uL15	4YBB	CM	all	prots1.pdb	J
uL16	4YBB	CN	all	prots1.pdb	K
bL17	4YBB	CO	all	prots1.pdb	L
uL18	4YBB	CP	all	prots1.pdb	M
bL19	4YBB	CQ	all	prots1.pdb	N
bL20	4YBB	CR	all	prots1.pdb	O
bL21	4YBB	CS	all	prots1.pdb	P
uL22	4YBB	CT	all	prots1.pdb	Q
uL23	4YBB	CU	all	prots1.pdb	R
uL24	4YBB	CV	all	prots1.pdb	S
bL25	4YBB	CW	all	prots1.pdb	T
bL27	4YBB	CX	all	prots1.pdb	U
bL28	4YBB	CY	all	prots1.pdb	V

uL29	4YBB	CZ	all	prots1.pdb	W
uL30	4YBB	C0	all	prots1.pdb	X
bL32	4YBB	C1	all	prots1.pdb	Y
bL33	4YBB	C2	all	prots1.pdb	Z
bL34	4YBB	C3	all	prots2.pdb	A
bL35	4YBB	C4	all	prots2.pdb	B
bL36	4YBB	C5	all	prots2.pdb	C

Supplementary Table 1: Residue and chain assignments for subunit occupancy analysis. Ribosomal RNA helices and ribosomal proteins are numbered as in Ref. 16.

Class	Centroid index
1	51011
2	51189
3	80371
4	9177
5	74182
6	73537
7	95314
8	66910
9	53298
10	81097
11	11144
12	71755
13	46961
14	37896
15	89122

Supplementary Table 2: Particle stack indices for the centroid volume of each subunit occupancy class. Note that these indices are only relevant for the provided pre-computed results and users should select alternative indices when training new cryoDRGN models.