pgn_export

November 8, 2022

```python
[3]: # PROTEIN GRAPH NETWORK: EXPORT ATOM LIST
import pandas as pd
import os

def export_atom_list(atom_list,output_path):
    print('\n[running export_atom_list]')

    # check output path
    if type(output_path)!=str or ' ' in output_path:
        raise Exception(f' ERROR: error reading output path: [{output_path}];␣
 ↪must be a string without spaces')
    try:
        if not os.path.exists(output_path):
            os.makedirs(output_path)
            print(f' NOTE: creating output path: [{output_path}]')
    except:
        raise Exception(f' ERROR: error reading output path: [{output_path}]')

    # save file
    file_path = f"{output_path}/atom_list.csv"
    df = pd.DataFrame(atom_list)
    df.columns = ['index','atom_number','atom_name','atom_type',
                  ␣
 ↪'residue_number','residue_name','chain_id','x_coord','y_coord',
                  'z_coord','element']
    df.to_csv(file_path,index=False)
    print(f' NOTE: saving atom list to output path: [{file_path}]')


# PROTEIN GRAPH NETWORK: EXPORT DISTANCE MATRIX
import numpy as np
import os

def export_distance_matrix(dist_matrix,output_path):
    print('\n[running export_distance_matrix]')

    # check output path
```

```python
    if type(output_path)!=str or ' ' in output_path:
        raise Exception(f' ERROR: error reading output path: [{output_path}];␣
 ↪must be a string without spaces')
    try:
        if not os.path.exists(output_path):
            os.makedirs(output_path)
            print(f' NOTE: creating output path: [{output_path}]')
    except:
        raise Exception(f' ERROR: error reading output path: [{output_path}]')

    # save file
    file_path = f"{output_path}/dist_matrix.csv"
    np.savetxt(file_path,dist_matrix,delimiter=',',fmt='%.18e')
    print(f' NOTE: saving distance matrix to output path: [{file_path}]')


# PROTEIN GRAPH NETWORK: EXPORT CONNECTION MATRIX
import csv
import os

def export_connection_matrix(conn_matrix,output_path):
    print('\n[running export_connection_matrix]')

    # check output path
    if type(output_path)!=str or ' ' in output_path:
        raise Exception(f' ERROR: error reading output path: [{output_path}];␣
 ↪must be a string without spaces')
    try:
        if not os.path.exists(output_path):
            os.makedirs(output_path)
            print(f' NOTE: creating output path: [{output_path}]')
    except:
        raise Exception(f' ERROR: error reading output path: [{output_path}]')

    # save file
    file_path = f"{output_path}/conn_matrix.csv"
    with open(file_path,'w',newline='') as file:
            mywriter = csv.writer(file, delimiter=',')
            mywriter.writerows(conn_matrix)
    print(f' NOTE: saving connection matrix to output path: [{file_path}]')


# PROTEIN GRAPH NETWORK: EXPORT BOND LIST
import pandas as pd
import os

def export_bond_list(bond_list,output_path):
```

```python
    print('\n[running export_bond_list]')

    # check output path
    if type(output_path)!=str or ' ' in output_path:
        raise Exception(f' ERROR: error reading output path: [{output_path}];␣
↪must be a string without spaces')
    try:
        if not os.path.exists(output_path):
            os.makedirs(output_path)
            print(f' NOTE: creating output path: [{output_path}]')
    except:
        raise Exception(f' ERROR: error reading output path: [{output_path}]')

    # save file
    file_path = f"{output_path}/bond_list.csv"
    df = pd.DataFrame(bond_list)
    df.columns = ['bond_index','a_index','b_index','bond_type','bond_length',
                  ␣
↪'a_atom_number','a_atom_name','a_atom_type','a_residue_number',
                  ␣
↪'a_residue_name','a_chain_id','a_x_coord','a_y_coord','a_z_coord',
                  'a_element',
                  ␣
↪'b_atom_number','b_atom_name','b_atom_type','b_residue_number',
                  ␣
↪'b_residue_name','b_chain_id','b_x_coord','b_y_coord','b_z_coord',
                  'b_element']
    df.to_csv(file_path,index=False)
    print(f' NOTE: saving bond list to output path: [{file_path}]')
```

[ ]: