*Submitted to CrystEngComm*

# Molecular-level insights of small organic molecule dipyrone crystallization by MD-based strategies

*Yi Sui* [a], *Wenchun Jiang* [a, b,] *, *Yingzheng Meng* [b], *Zhuwen Shao* [b ,c], *Huibo Meng* [b,] *

[a] School of Petroleum Engineering, State Key Laboratory of Heavy Oil Processing, China University of Petroleum (East China), Qingdao 266580, P.R. China

[b] College of New Energy, State Key Laboratory of Heavy Oil Processing, China University of Petroleum (East China), Qingdao 266580, P.R. China

[c] College of Electromechanical Engineering, Qingdao University of Science and Technology, Qingdao 266061, P.R. China

* Corresponding author, E-mail: jiangwenchun@126.com (W. Jiang); huibomeng@126.com (H. Meng)

Table S1. Atom type and the atom charges of the Restrained Electrostatic Potential (RESP).

| Atom No. | Atom index | Atom type | Atom charge | Atom mass |
|---|---|---|---|---|
| 1 | N1 | n | -0.2072 | 14.0067 |
| 2 | N2 | na | -0.20837 | 14.0067 |
| 3 | C3 | cc | 0.103756 | 12.01074 |
| 4 | C4 | cc | -0.03843 | 12.01074 |
| 5 | C5 | c | 0.575799 | 12.01074 |
| 6 | C6 | c3 | -0.10902 | 12.01074 |
| 7 | C7 | c3 | -0.30428 | 12.01074 |
| 8 | O8 | o | -0.61416 | 15.99941 |
| 9 | N9 | nh | -0.10324 | 14.0067 |
| 10 | C10 | c3 | -0.40563 | 12.01074 |
| 11 | C11 | c3 | -0.24019 | 12.01074 |
| 12 | S12 | s6 | 0.976247 | 32.06479 |
| 13 | O13 | o | -0.6373 | 15.99941 |
| 14 | O14 | o | -0.67735 | 15.99941 |
| 15 | O15 | o | -0.6078 | 15.99941 |
| 16 | C16 | ca | 0.300484 | 12.01074 |
| 17 | C17 | ca | -0.20604 | 12.01074 |
| 18 | C18 | ca | -0.11628 | 12.01074 |
| 19 | C19 | ca | -0.18898 | 12.01074 |
| 20 | C20 | ca | -0.11125 | 12.01074 |
| 21 | C21 | ca | -0.25108 | 12.01074 |
| 22 | H22 | h1 | 0.101548 | 1.007941 |
| 23 | H23 | h1 | 0.101548 | 1.007941 |
| 24 | H24 | h1 | 0.101548 | 1.007941 |
| 25 | H25 | hc | 0.123915 | 1.007941 |
| 26 | H26 | hc | 0.123915 | 1.007941 |
| 27 | H27 | hc | 0.123915 | 1.007941 |
| 28 | H28 | h1 | 0.139866 | 1.007941 |
| 29 | H29 | h1 | 0.139866 | 1.007941 |
| 30 | H30 | h1 | 0.139866 | 1.007941 |
| 31 | H31 | h2 | 0.139533 | 1.007941 |
| 32 | H32 | h2 | 0.139533 | 1.007941 |
| 33 | H33 | ha | 0.13102 | 1.007941 |
| 34 | H34 | ha | 0.133382 | 1.007941 |
| 35 | H35 | ha | 0.138541 | 1.007941 |
| 36 | H36 | ha | 0.132591 | 1.007941 |
| 37 | H37 | ha | 0.159745 | 1.007941 |

Table S2. Bond information.

| Bond type | Bond length (nm) | *k* (kJ/mol/nm^2) |
|---|---|---|
| N1-N2 | 0.14071 | 3.72E+05 |
| N1-C5 | 0.13789 | 3.58E+05 |
| N1-C16 | 0.14121 | 3.21E+05 |
| N2-C3 | 0.13802 | 3.56E+05 |
| N2-C6 | 0.14629 | 2.74E+05 |
| C3-C4 | 0.14278 | 3.51E+05 |
| C3-C7 | 0.15015 | 2.80E+05 |
| C4-C5 | 0.14676 | 3.10E+05 |
| C4-N9 | 0.13735 | 3.64E+05 |
| C5-O8 | 0.12183 | 5.34E+05 |
| C6-H22 | 0.10969 | 2.77E+05 |
| C6-H23 | 0.10969 | 2.77E+05 |
| C6-H24 | 0.10969 | 2.77E+05 |
| C7-H25 | 0.10969 | 2.77E+05 |
| C7-H26 | 0.10969 | 2.77E+05 |
| C7-H27 | 0.10969 | 2.77E+05 |
| N9-C10 | 0.1464 | 2.73E+05 |
| N9-C11 | 0.1464 | 2.73E+05 |
| C10-H28 | 0.10969 | 2.77E+05 |
| C10-H29 | 0.10969 | 2.77E+05 |
| C10-H30 | 0.10969 | 2.77E+05 |
| C11-S12 | 0.18075 | 1.95E+05 |
| C11-H31 | 0.10961 | 2.78E+05 |
| C11-H32 | 0.10961 | 2.78E+05 |
| S12-O13 | 0.14533 | 4.29E+05 |
| S12-O14 | 0.14533 | 4.29E+05 |
| S12-O15 | 0.14533 | 4.29E+05 |
| C16-C17 | 0.13984 | 3.86E+05 |
| C16-C21 | 0.13984 | 3.86E+05 |
| C17-C18 | 0.13984 | 3.86E+05 |
| C17-H33 | 0.1086 | 2.89E+05 |
| C18-C19 | 0.13984 | 3.86E+05 |
| C18-H34 | 0.1086 | 2.89E+05 |
| C19-C20 | 0.13984 | 3.86E+05 |
| C19-H35 | 0.1086 | 2.89E+05 |
| C20-C21 | 0.13984 | 3.86E+05 |
| C20-H36 | 0.1086 | 2.89E+05 |
| C21-H37 | 0.1086 | 2.89E+05 |

Table S3. Angle information.

| Angle type | Angle (degree) | *k* (kJ/mol/rad^2) |
|---|---|---|
| N2-N1-C5 | 111.5 | 5.77E+02 |
| N2-N1-C16 | 119.3 | 5.51E+02 |
| C5-N1-C16 | 123.71 | 5.34E+02 |
| N1-N2-C3 | 110.05 | 5.81E+02 |
| N1-N2-C6 | 112.88 | 5.56E+02 |
| C3-N2-C6 | 126.46 | 5.18E+02 |
| N2-C3-C4 | 117.77 | 5.74E+02 |
| N2-C3-C7 | 122.73 | 5.46E+02 |
| C4-C3-C7 | 115.97 | 5.41E+02 |
| C3-C4-C5 | 122.69 | 5.32E+02 |
| C3-C4-N9 | 119.72 | 5.71E+02 |
| C5-C4-N9 | 125.703 | 5.00E+02 |
| N1-C5-C4 | 112.7 | 5.78E+02 |
| N1-C5-O8 | 123.05 | 6.21E+02 |
| C4-C5-O8 | 123.93 | 5.78E+02 |
| N2-C6-H22 | 108.78 | 4.17E+02 |
| N2-C6-H23 | 108.78 | 4.17E+02 |
| N2-C6-H24 | 108.78 | 4.17E+02 |
| H22-C6-H23 | 108.46 | 3.28E+02 |
| H22-C6-H24 | 108.46 | 3.28E+02 |
| H23-C6-H24 | 108.46 | 3.28E+02 |
| C3-C7-H25 | 110.49 | 3.95E+02 |
| C3-C7-H26 | 110.49 | 3.95E+02 |
| C3-C7-H27 | 110.49 | 3.95E+02 |
| H25-C7-H26 | 107.58 | 3.30E+02 |
| H25-C7-H27 | 107.58 | 3.30E+02 |
| H26-C7-H27 | 107.58 | 3.30E+02 |
| C4-N9-C10 | 119.72 | 5.33E+02 |
| C4-N9-C11 | 119.72 | 5.33E+02 |
| C10-N9-C11 | 114.51 | 5.29E+02 |
| N9-C10-H28 | 109.79 | 4.15E+02 |
| N9-C10-H29 | 109.79 | 4.15E+02 |
| N9-C10-H30 | 109.79 | 4.15E+02 |
| H28-C10-H29 | 108.46 | 3.28E+02 |
| H28-C10-H30 | 108.46 | 3.28E+02 |
| H29-C10-H30 | 108.46 | 3.28E+02 |
| N9-C11-S12 | 115.571 | 5.00E+02 |
| N9-C11-H31 | 110.01 | 4.14E+02 |
| N9-C11-H32 | 110.01 | 4.14E+02 |

|  |  |  |
|---|---|---|
| S12-C11-H31 | 106.51 | 3.62E+02 |
| S12-C11-H32 | 106.51 | 3.62E+02 |
| H31-C11-H32 | 110.2 | 3.26E+02 |
| C11-S12-O13 | 108.61 | 5.47E+02 |
| C11-S12-O14 | 108.61 | 5.47E+02 |
| C11-S12-O15 | 108.61 | 5.47E+02 |
| O13-S12-O14 | 120.05 | 6.16E+02 |
| O13-S12-O15 | 120.05 | 6.16E+02 |
| O14-S12-O15 | 120.05 | 6.16E+02 |
| N1-C16-C17 | 120.19 | 5.68E+02 |
| N1-C16-C21 | 120.19 | 5.68E+02 |
| C17-C16-C21 | 120.02 | 5.57E+02 |
| C16-C17-C18 | 120.02 | 5.57E+02 |
| C16-C17-H33 | 119.88 | 4.03E+02 |
| C18-C17-H33 | 119.88 | 4.03E+02 |
| C17-C18-C19 | 120.02 | 5.57E+02 |
| C17-C18-H34 | 119.88 | 4.03E+02 |
| C19-C18-H34 | 119.88 | 4.03E+02 |
| C18-C19-C20 | 120.02 | 5.57E+02 |
| C18-C19-H35 | 119.88 | 4.03E+02 |
| C20-C19-H35 | 119.88 | 4.03E+02 |
| C19-C20-C21 | 120.02 | 5.57E+02 |
| C19-C20-H36 | 119.88 | 4.03E+02 |
| C21-C20-H36 | 119.88 | 4.03E+02 |
| C16-C21-C20 | 120.02 | 5.57E+02 |
| C16-C21-H37 | 119.88 | 4.03E+02 |
| C20-C21-H37 | 119.88 | 4.03E+02 |
| N2-N1-C5 | 111.5 | 5.77E+02 |
| N2-N1-C16 | 119.3 | 5.51E+02 |
| C5-N1-C16 | 123.71 | 5.34E+02 |

Table S4. Dihedrals (propers).

| Dihedral type | Phase (degree) | $k_d$ (kJ/mol) | $p_n$ |
|---|---|---|---|
| N1-N2-C3-C4 | 180 | 7.1128 | 2 |
| N1-N2-C3-C7 | 180 | 7.1128 | 2 |
| N1-N2-C6-H22 | 0 | 0 | 2 |
| N1-N2-C6-H23 | 0 | 0 | 2 |
| N1-N2-C6-H24 | 0 | 0 | 2 |
| N1-C5-C4-C3 | 180 | 12.029 | 2 |
| N1-C5-C4-N9 | 180 | 12.029 | 2 |

| | | | |
|---|---|---|---|
| N1-C16-C17-C18 | 180 | 15.167 | 2 |
| N1-C16-C17-H33 | 180 | 15.167 | 2 |
| N1-C16-C21-C20 | 180 | 15.167 | 2 |
| N1-C16-C21-H37 | 180 | 15.167 | 2 |
| N2-N1-C5-C4 | 180 | 10.46 | 2 |
| N2-N1-C5-O8 | 180 | 10.46 | 2 |
| N2-N1-C16-C17 | 180 | 1.8828 | 2 |
| N2-N1-C16-C21 | 180 | 1.8828 | 2 |
| N2-C3-C4-C5 | 180 | 16.736 | 2 |
| N2-C3-C4-N9 | 180 | 16.736 | 2 |
| N2-C3-C7-H25 | 0 | 0 | 3 |
| N2-C3-C7-H26 | 0 | 0 | 3 |
| N2-C3-C7-H27 | 0 | 0 | 3 |
| C3-N2-N1-C5 | 0 | 2.9288 | 2 |
| C3-N2-N1-C16 | 0 | 2.9288 | 2 |
| C3-N2-C6-H22 | 0 | 0 | 2 |
| C3-N2-C6-H23 | 0 | 0 | 2 |
| C3-N2-C6-H24 | 0 | 0 | 2 |
| C3-C4-C5-O8 | 180 | 12.029 | 2 |
| C3-C4-N9-C10 | 180 | 4.3932 | 2 |
| C3-C4-N9-C11 | 180 | 4.3932 | 2 |
| C4-C3-N2-C6 | 180 | 7.1128 | 2 |
| C4-C3-C7-H25 | 0 | 0 | 3 |
| C4-C3-C7-H26 | 0 | 0 | 3 |
| C4-C3-C7-H27 | 0 | 0 | 3 |
| C4-C5-N1-C16 | 180 | 10.46 | 2 |
| C4-N9-C10-H28 | 0 | 0 | 2 |
| C4-N9-C10-H29 | 0 | 0 | 2 |
| C4-N9-C10-H30 | 0 | 0 | 2 |
| C4-N9-C11-S12 | 0 | 0 | 2 |
| C4-N9-C11-H31 | 0 | 0 | 2 |
| C4-N9-C11-H32 | 0 | 0 | 2 |
| C5-N1-N2-C6 | 0 | 2.9288 | 2 |
| C5-N1-C16-C17 | 180 | 1.8828 | 2 |
| C5-N1-C16-C21 | 180 | 1.8828 | 2 |
| C5-C4-C3-C7 | 180 | 16.736 | 2 |
| C5-C4-N9-C10 | 180 | 4.3932 | 2 |
| C5-C4-N9-C11 | 180 | 4.3932 | 2 |
| C6-N2-N1-C16 | 0 | 2.9288 | 2 |
| C6-N2-C3-C7 | 180 | 7.1128 | 2 |
| C7-C3-C4-N9 | 180 | 16.736 | 2 |

| | | | |
|---|---|---|---|
| O8-C5-N1-C16 | 180 | 10.46 | 2 |
| O8-C5-C4-N9 | 180 | 12.029 | 2 |
| N9-C11-S12-O13 | 0 | 0.60436 | 3 |
| N9-C11-S12-O14 | 0 | 0.60436 | 3 |
| N9-C11-S12-O15 | 0 | 0.60436 | 3 |
| C10-N9-C11-S12 | 0 | 0 | 2 |
| C10-N9-C11-H31 | 0 | 0 | 2 |
| C10-N9-C11-H32 | 0 | 0 | 2 |
| C11-N9-C10-H28 | 0 | 0 | 2 |
| C11-N9-C10-H29 | 0 | 0 | 2 |
| C11-N9-C10-H30 | 0 | 0 | 2 |
| O13-S12-C11-H31 | 0 | 0.60436 | 3 |
| O13-S12-C11-H32 | 0 | 0.60436 | 3 |
| O14-S12-C11-H31 | 0 | 0.60436 | 3 |
| O14-S12-C11-H32 | 0 | 0.60436 | 3 |
| O15-S12-C11-H31 | 0 | 0.60436 | 3 |
| O15-S12-C11-H32 | 0 | 0.60436 | 3 |
| C16-C17-C18-C19 | 180 | 15.167 | 2 |
| C16-C17-C18-H34 | 180 | 15.167 | 2 |
| C16-C21-C20-C19 | 180 | 15.167 | 2 |
| C16-C21-C20-H36 | 180 | 15.167 | 2 |
| C17-C16-C21-C20 | 180 | 15.167 | 2 |
| C17-C16-C21-H37 | 180 | 15.167 | 2 |
| C17-C18-C19-C20 | 180 | 15.167 | 2 |
| C17-C18-C19-H35 | 180 | 15.167 | 2 |
| C18-C17-C16-C21 | 180 | 15.167 | 2 |
| C18-C19-C20-C21 | 180 | 15.167 | 2 |
| C18-C19-C20-H36 | 180 | 15.167 | 2 |
| C19-C18-C17-H33 | 180 | 15.167 | 2 |
| C19-C20-C21-H37 | 180 | 15.167 | 2 |
| C20-C19-C18-H34 | 180 | 15.167 | 2 |
| C21-C16-C17-H33 | 180 | 15.167 | 2 |
| C21-C20-C19-H35 | 180 | 15.167 | 2 |
| H33-C17-C18-H34 | 180 | 15.167 | 2 |
| H34-C18-C19-H35 | 180 | 15.167 | 2 |
| H35-C19-C20-H36 | 180 | 15.167 | 2 |
| H36-C20-C21-H37 | 180 | 15.167 | 2 |

Table S5. Dihedrals (impropers).

| Dihedral type | Phase (degree) | $k_d$ (kJ/mol) | $p_n$ |
|---|---|---|---|

| | | | |
|---|---|---|---|
| N2-C5-N1-C16 | 180 | 4.6024 | 2 |
| N1-C3-N2-C6 | 180 | 4.6024 | 2 |
| N1-C4-C5-O8 | 180 | 43.932 | 2 |
| N1-C17-C16-C21 | 180 | 4.6024 | 2 |
| C16-C18-C17-H33 | 180 | 4.6024 | 2 |
| C17-C19-C18-H34 | 180 | 4.6024 | 2 |
| C18-C20-C19-H35 | 180 | 4.6024 | 2 |
| C19-C21-C20-H36 | 180 | 4.6024 | 2 |
| C16-C20-C21-H37 | 180 | 4.6024 | 2 |

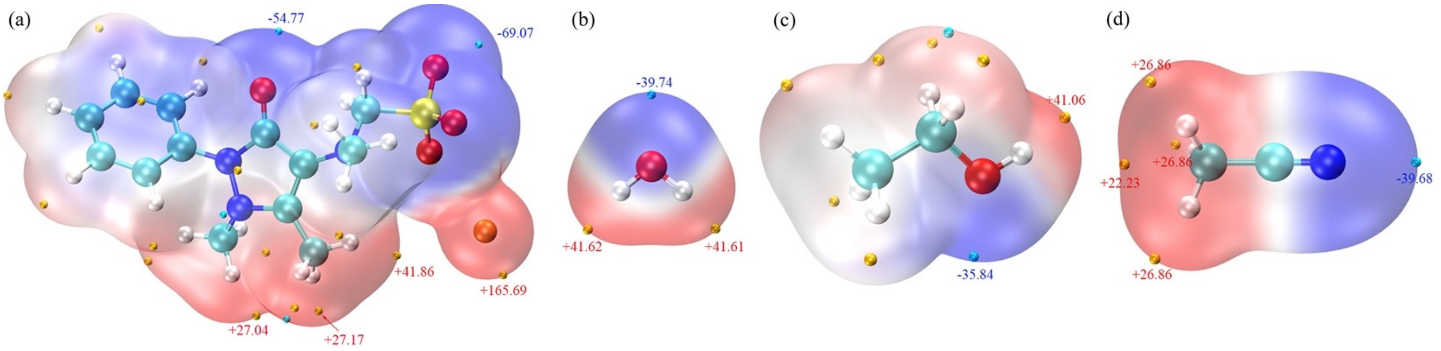

Figure S1. Molecular surface analysis over van der Waals surface colored by electrostatic potential (ESP) where cyan and yellow spheres in the maps highlight the surface ESP minima and maxima. The four molecules mentioned in the paper: (a) dipyrone; (b) water; (c) ethanol; (d) acetonitrile.

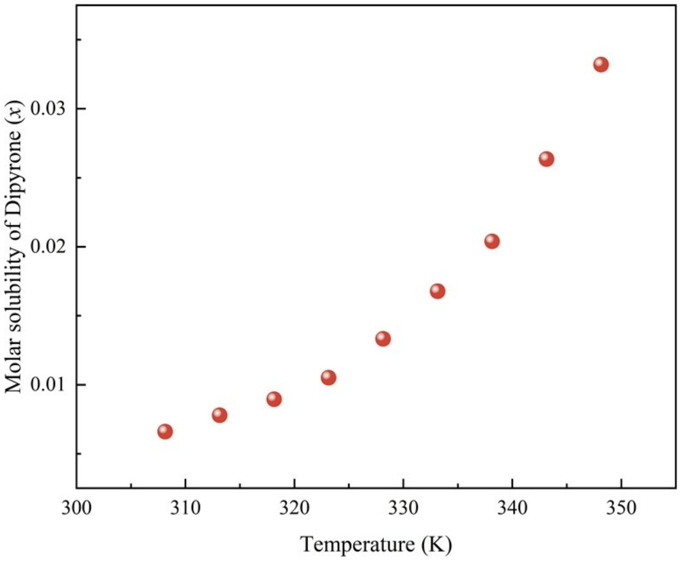

Figure S2. Mole fraction solubility of dipyrone in 95% ethanol at different temperature

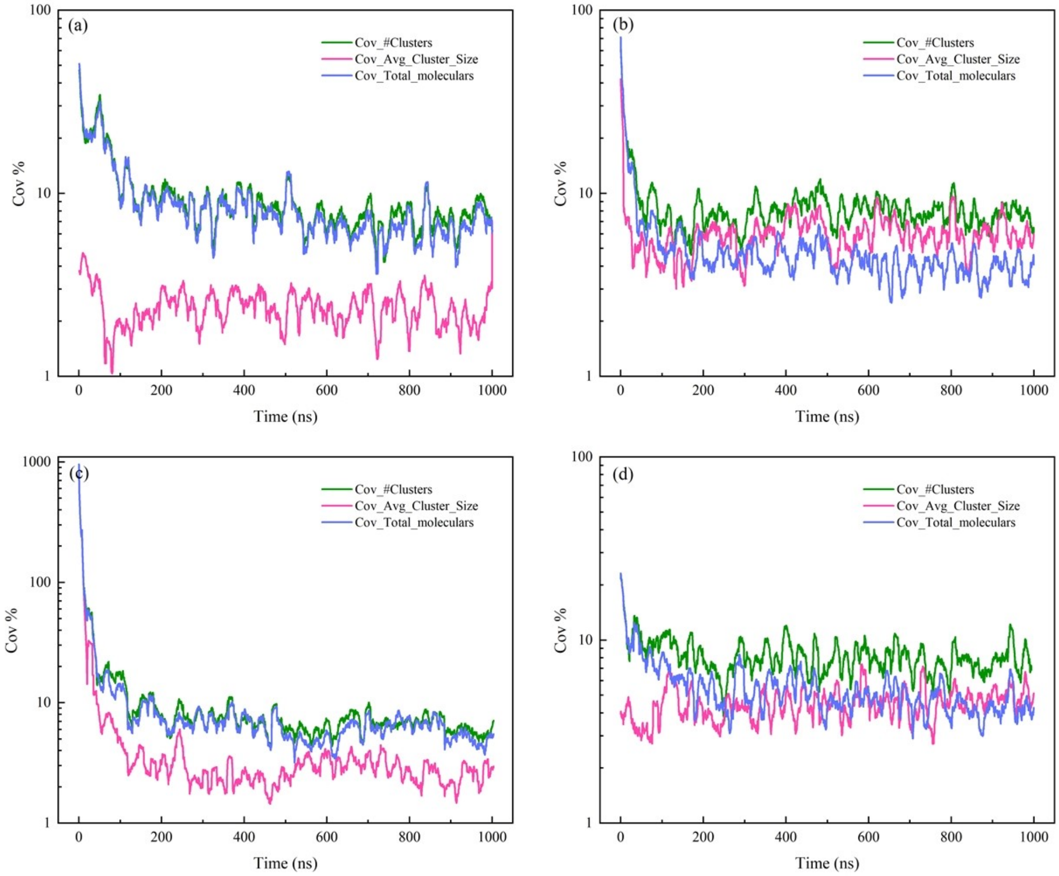

Fig. S3. Time-dependent curves of the coefficient of variation for the number, average size, and total number of dipyrone clusters formed in different solutions: (a) EtOH; (b) 95% EtOH; (c) ACN; (d) 95% ACN.

30

31

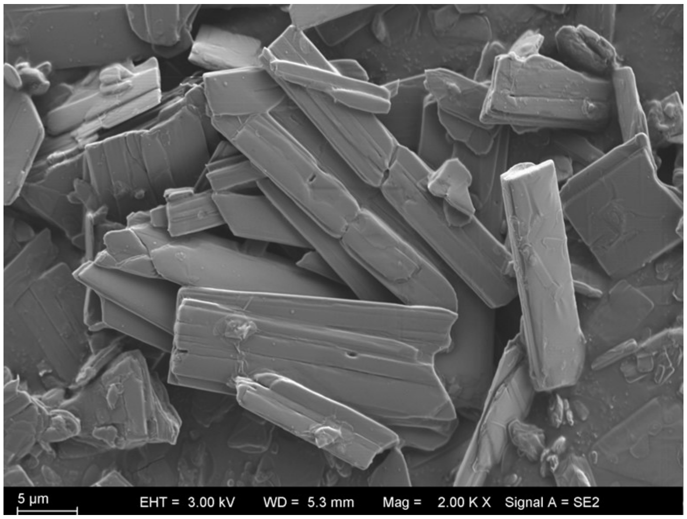

Figure S4. SEM image of the dipyrone crystal in 95% EtOH.

32

33 **Tcl code for molecular classification in Fig.3a:**

```
set range1_start 1
set range1_end 480
set range2_start 1441
set range2_end 1920
set start_frame 0
set end_frame 405
set displacement_threshold 4.5
set angle_change_threshold 50.0
set neighbor_distance 13.5
set z_min 42.0
set z_max 145.0

# set num_reps [molinfo top get numreps]
# for {set i 0} {$i < $num_reps} {incr i} {
#       mol delrep 0 top
# }

set outfile [open "frame_stats_prod3-50degree-13dot5.csv" w]
puts $outfile "frame,crystal_count,A_molecule_count,solu_count"

for {set frame $start_frame} {$frame < $end_frame} {incr frame} {
    set current_frame $frame
    set next_frame [expr {$frame + 1}]

    puts "Analyzing frames $current_frame to $next_frame..."

    array unset res_coords_frame0
    array unset res_coords_frame1
    array unset res_angles
    array unset res_displacement
    array unset res_class

    foreach {type start end} [list range1 $range1_start $range1_end range2 $range2_start $range2_end]
{
        for {set ires $start} {$ires <= $end} {incr ires} {
            set sel_N1 [atomselect top "resid $ires and name N1" frame $current_frame]
            set sel_N9 [atomselect top "resid $ires and name N9" frame $current_frame]

            if {[$sel_N1 num] != 1 || [$sel_N9 num] != 1} {
                $sel_N1 delete
                $sel_N9 delete
                continue
            }

            set coords_N1 [$sel_N1 get {x y z}]
            set coords_N9 [$sel_N9 get {x y z}]

            if {[llength $coords_N1] != 1 || [llength $coords_N9] != 1} {
                $sel_N1 delete
                $sel_N9 delete
                continue
            }

            set x1 [lindex $coords_N1 0 0]
            set y1 [lindex $coords_N1 0 1]
            set z1 [lindex $coords_N1 0 2]
            set x9 [lindex $coords_N9 0 0]
            set y9 [lindex $coords_N9 0 1]
            set z9 [lindex $coords_N9 0 2]

            if {[string is double $x1] && [string is double $y1] && [string is double $z1] &&
                [string is double $x9] && [string is double $y9] && [string is double $z9]} {

                set res_coords_frame0($ires) [list $x1 $y1 $z1 $x9 $y9 $z9]
            }

            $sel_N1 delete
            $sel_N9 delete
        }
    }

    foreach {type start end} [list range1 $range1_start $range1_end range2 $range2_start $range2_end]
{
        for {set ires $start} {$ires <= $end} {incr ires} {
            set sel_N1 [atomselect top "resid $ires and name N1" frame $next_frame]
            set sel_N9 [atomselect top "resid $ires and name N9" frame $next_frame]

            if {[$sel_N1 num] != 1 || [$sel_N9 num] != 1} {
                $sel_N1 delete
                $sel_N9 delete
                continue
            }

            set coords_N1 [$sel_N1 get {x y z}]
            set coords_N9 [$sel_N9 get {x y z}]

            if {[llength $coords_N1] != 1 || [llength $coords_N9] != 1} {
                $sel_N1 delete
                $sel_N9 delete
                continue
            }

            set x1 [lindex $coords_N1 0 0]
            set y1 [lindex $coords_N1 0 1]
            set z1 [lindex $coords_N1 0 2]
            set x9 [lindex $coords_N9 0 0]
            set y9 [lindex $coords_N9 0 1]
            set z9 [lindex $coords_N9 0 2]

            if {[string is double $x1] && [string is double $y1] && [string is double $z1] &&
                [string is double $x9] && [string is double $y9] && [string is double $z9]} {

                set res_coords_frame1($ires) [list $x1 $y1 $z1 $x9 $y9 $z9]
            }

            $sel_N1 delete
            $sel_N9 delete
        }
    }

    set molid [molinfo top]

    set cell [pbc get -molid $molid -first $current_frame -last $current_frame]

    if {[llength $cell] >= 6} {
        set a [lindex $cell 0]
        set b [lindex $cell 1]
        set c [lindex $cell 2]
        set alpha [lindex $cell 3]
        set beta [lindex $cell 4]
        set gamma [lindex $cell 5]
    } else {
        set a 50.0
        set b 100.0
        set c 180.0
        set alpha 90.0
        set beta 90.0
        set gamma 90.0
    }

    set tolerance 0.1
    if {abs($alpha-90.0)>$tolerance || abs($beta-90.0)>$tolerance || abs($gamma-90.0)>$tolerance} {
        puts "Warning: Non-orthogonal box detected (angles: $alpha, $beta, $gamma). The PBC correction may not be accurate for non-orthogonal boxes. This script currently only supports orthogonal boxes."
    }

    proc pbc_correct_coords {x0 y0 z0 x1 y1 z1 a b c} {
        set dx [expr {$x1 - $x0}]
        set dy [expr {$y1 - $y0}]
        set dz [expr {$z1 - $z0}]

        if {abs($dx) > $a/2.0} {
            if {$dx > 0} {
                set x1 [expr {$x1 - $a}]
            } else {
                set x1 [expr {$x1 + $a}]
            }
        }

        if {abs($dy) > $b/2.0} {
            if {$dy > 0} {
                set y1 [expr {$y1 - $b}]
            } else {
                set y1 [expr {$y1 + $b}]
            }
        }

        if {abs($dz) > $c/2.0} {
            if {$dz > 0} {
                set z1 [expr {$z1 - $c}]
            } else {
                set z1 [expr {$z1 + $c}]
            }
        }

        return [list $x1 $y1 $z1]
    }

    proc pbc_distance {coord1 coord2 a b c} {
        lassign $coord1 x1 y1 z1
        lassign $coord2 x2 y2 z2

        set dx [expr {$x2 - $x1}]
        set dy [expr {$y2 - $y1}]
        set dz [expr {$z2 - $z1}]

        if {abs($dx) > $a/2.0} {
            if {$dx > 0} {
                set dx [expr {$dx - $a}]
            } else {
                set dx [expr {$dx + $a}]
            }
        }

        if {abs($dy) > $b/2.0} {
            if {$dy > 0} {
                set dy [expr {$dy - $b}]
            } else {
                set dy [expr {$dy + $b}]
            }
        }

        if {abs($dz) > $c/2.0} {
            if {$dz > 0} {
                set dz [expr {$dz - $c}]
            } else {
                set dz [expr {$dz + $c}]
            }
        }

        return [expr {sqrt($dx*$dx + $dy*$dy + $dz*$dz)}]
    }

```

```
proc angle_with_y_axis {dx dy dz} {
    set magnitude [expr {sqrt($dx*$dx + $dy*$dy + $dz*$dz)}]
    if {$magnitude < 0.0001} {
        return 0.0
    }

    set dot_product [expr {$dy / $magnitude}]

    if {$dot_product > 1.0} {set dot_product 1.0}
    if {$dot_product < -1.0} {set dot_product -1.0}

    set angle [expr {acos($dot_product) * 180.0 / 3.14159265359}]
    return $angle
}

proc vector_angle_between {v1 v2} {
    lassign $v1 dx1 dy1 dz1
    lassign $v2 dx2 dy2 dz2

    set dot_product [expr {$dx1*$dx2 + $dy1*$dy2 + $dz1*$dz2}]
    set mag1 [expr {sqrt($dx1*$dx1 + $dy1*$dy1 + $dz1*$dz1)}]
    set mag2 [expr {sqrt($dx2*$dx2 + $dy2*$dy2 + $dz2*$dz2)}]

    if {$mag1 < 0.0001 || $mag2 < 0.0001} {
        return 0.0
    }

    set cos_theta [expr {$dot_product / ($mag1 * $mag2)}]
    if {$cos_theta > 1.0} {set cos_theta 1.0}
    if {$cos_theta < -1.0} {set cos_theta -1.0}

    set angle [expr {acos($cos_theta) * 180.0 / 3.14159265359}]
    return $angle
}

set pre_crystal_resids [list]
set disorder_resids [list]

foreach res [array names res_coords_frame0] {
    if {![info exists res_coords_frame1($res)]} {
        continue
        }

        lassign $res_coords_frame0($res) x1_0 y1_0 z1_0 x9_0 y9_0 z9_0

        #
        if {($res >= $range1_start && $res <= $range1_end) || ($res >= $range2_start && $res <=
$range2_end)} {
            #
            lassign $res_coords_frame1($res) x1_1 y1_1 z1_1 x9_1 y9_1 z9_1
            lassign [pbc_correct_coords $x9_0 $y9_0 $z9_0 $x9_1 $y9_1 $z9_1 $a $b $c] x9_1_corr
y9_1_corr z9_1_corr
            set dx [expr {$x9_1_corr - $x9_0}]
            set dy [expr {$y9_1_corr - $y9_0}]
            set dz [expr {$z9_1_corr - $z9_0}]
            set displacement [expr {sqrt($dx*$dx + $dy*$dy + $dz*$dz)}]
            set res_displacement($res) $displacement

            #
            set dx0 [expr {$x9_0 - $x1_0}]
            set dy0 [expr {$y9_0 - $y1_0}]
            set dz0 [expr {$z9_0 - $z1_0}]
            set vector0 [list $dx0 $dy0 $dz0]

            #
            set dx1 [expr {$x9_1 - $x1_1}]
            set dy1 [expr {$y9_1 - $y1_1}]
            set dz1 [expr {$z9_1 - $z1_1}]
            set vector1 [list $dx1 $dy1 $dz1]

            #
            set angle0 [angle_with_y_axis $dx0 $dy0 $dz0]

            #
            set rotation_angle [vector_angle_between $vector0 $vector1]
            set res_angles($res) [list $angle0 $rotation_angle]

            if {$displacement <= $displacement_threshold &&
                $rotation_angle <= $angle_change_threshold &&
                (($angle0 >= 0.0 && $angle0 <= 50.0) || ($angle0 >= 130.0 && $angle0 <= 180.0))}
{
                lappend pre_crystal_resids $res
                set res_class($res) "pre_crystal"
            } else {
                lappend disorder_resids $res
                set res_class($res) "disorder"
            }
        }
    }

    set crystal_resids [list]
    set solu_resids [list]

    foreach res $pre_crystal_resids {
        lassign $res_coords_frame0($res) x1_0 y1_0 z1_0 x9_0 y9_0 z9_0
        set coord1 [list $x1_0 $y1_0 $z1_0]

        set neighbor_count 0

        foreach other_res $pre_crystal_resids {
            if {$res == $other_res} continue

            lassign $res_coords_frame0($other_res) x1_other y1_other z1_other x9_other y9_other z9_other
            set coord2 [list $x1_other $y1_other $z1_other]

            set dist [pbc_distance $coord1 $coord2 $a $b $c]

            if {$dist <= $neighbor_distance} {
                incr neighbor_count
            }
        }

        if {$neighbor_count >= 2} {
            lappend crystal_resids $res
            set res_class($res) "crystal"
        } else {
            lappend solu_resids $res
            set res_class($res) "solu"
        }
    }

    set changed 1
    set iteration 0
    while {$changed && $iteration < 5} {
        set changed 0
        set new_crystal_resids [list]
        set new_solu_resids [list]

        array set current_crystal_set {}
        foreach res $crystal_resids {
            set current_crystal_set($res) 1
        }

        foreach res $crystal_resids {
            lassign $res_coords_frame0($res) x1_0 y1_0 z1_0 x9_0 y9_0 z9_0
            set coord1 [list $x1_0 $y1_0 $z1_0]
            set neighbor_count 0
            foreach other_res $crystal_resids {
                if {$res == $other_res} continue
                if {![info exists current_crystal_set($other_res)]} continue
                lassign $res_coords_frame0($other_res) x1_other y1_other z1_other x9_other y9_other z9_other
                set coord2 [list $x1_other $y1_other $z1_other]
                set dist [pbc_distance $coord1 $coord2 $a $b $c]

                if {$dist <= $neighbor_distance} {
                    incr neighbor_count
                }
            }
            if {$neighbor_count >= 1} {
                lappend new_crystal_resids $res
            } else {
                lappend new_solu_resids $res
                set res_class($res) "solu"
                set changed 1
            }
        }
        set crystal_resids $new_crystal_resids
        set solu_resids [concat $solu_resids $new_solu_resids]

        incr iteration
    }

    set A_molecule_resids [list]
    foreach res $disorder_resids {
        if {[info exists res_coords_frame0($res)]} {
            lassign $res_coords_frame0($res) x1_0 y1_0 z1_0 x9_0 y9_0 z9_0
            if {$z1_0 < $z_min || $z1_0 > $z_max} {
                lappend crystal_resids $res
                set res_class($res) "crystal"
            }
        }
    }
    set new_disorder_resids [list]
    foreach res $disorder_resids {
        if {[info exists res_coords_frame0($res)]} {
            lassign $res_coords_frame0($res) x1_0 y1_0 z1_0 x9_0 y9_0 z9_0
            if {$z1_0 >= $z_min && $z1_0 <= $z_max} {
                lappend new_disorder_resids $res
            }
        }
    }
    set disorder_resids $new_disorder_resids
    foreach res $disorder_resids {
        lassign $res_coords_frame0($res) x1_0 y1_0 z1_0 x9_0 y9_0 z9_0
        set coord1 [list $x1_0 $y1_0 $z1_0]
        set has_crystal_neighbor 0
        foreach crystal_res $crystal_resids {
            lassign $res_coords_frame0($crystal_res) x1_crystal y1_crystal z1_crystal x9_crystal
y9_crystal z9_crystal
            set coord2 [list $x1_crystal $y1_crystal $z1_crystal]
            set dist [pbc_distance $coord1 $coord2 $a $b $c]
            if {$dist <= $neighbor_distance} {
                set has_crystal_neighbor 1
                break
            }
        }
        if {$has_crystal_neighbor} {
            lappend A_molecule_resids $res
            set res_class($res) "A_molecule"
        } else {
            lappend solu_resids $res
            set res_class($res) "solu"
        }
    }

    foreach res [array names res_coords_frame0] {
        if {[info exists res_class($res)]} {
            lassign $res_coords_frame0($res) x1_0 y1_0 z1_0 x9_0 y9_0 z9_0

            if {$z1_0 < $z_min || $z1_0 > $z_max} {
                if {$res_class($res) ne "crystal"} {
                    set idx [lsearch -exact $solu_resids $res]
                    if {$idx != -1} {
                        set solu_resids [lreplace $solu_resids $idx $idx]
                    }

                    set idx [lsearch -exact $A_molecule_resids $res]
                    if {$idx != -1} {
                        set A_molecule_resids [lreplace $A_molecule_resids $idx $idx]
                    }

                    set idx [lsearch -exact $disorder_resids $res]
                    if {$idx != -1} {
                        set disorder_resids [lreplace $disorder_resids $idx $idx]
                    }

                    lappend crystal_resids $res
                    set res_class($res) "crystal"
                }
            }
        }
    }

    #
    set crystal_count [llength $crystal_resids]
    set A_molecule_count [llength $A_molecule_resids]
    set solu_count [llength $solu_resids]

    #
    puts $outfile "$next_frame,$crystal_count,$A_molecule_count,$solu_count"

    puts "Frame $next_frame: Crystal=$crystal_count, A_molecule=$A_molecule_count, Solu=$solu_count"
}
```

```
close $outfile
puts "Multi-frame analysis completed. Results saved to frame_stats_0-3000-50degree-13dot5.csv"
```

**Tcl code for cluster statistics in Fig.3b:**

```
draw delete all

set range1_start 0
set range1_end 0
set range2_start 5723
set range2_end 5828
set crit 7
set output_file "connects.txt"
set min_cluster_size 3

set num_frames [molinfo top get numframes]

set outfile [open $output_file w]
puts $outfile "Frame\tYellow_Connects\tRed_Connects\tCluster_Count\tAvg_Cluster_Size"

for {set frame 0} {$frame < $num_frames} {incr frame} {
    animate goto $frame
    display update

    array unset yellow_connected
    array unset red_connected
    array unset cluster_assigned
    array unset cluster_size

    foreach {start end} [list $range1_start $range1_end $range3_start $range3_end] {
        for {set ires $start} {$ires <= $end} {incr ires} {
            set sel [atomselect top "resid $ires and name S12" frame $frame]
            if {[$sel num] > 0} {
                set coord [lindex [$sel get {x y z}] 0]
                set x($ires) [lindex $coord 0]
                set y($ires) [lindex $coord 1]
                set z($ires) [lindex $coord 2]
            }
            $sel delete
        }
    }
```

```
for {set ires $range2_start} {$ires <= $range2_end} {incr ires} {
    set sel [atomselect top "resid $ires and name S12" frame $frame]
    if {[$sel num] > 0} {
        set coord [lindex [$sel get {x y z}] 0]
        set x($ires) [lindex $coord 0]
        set y($ires) [lindex $coord 1]
        set z($ires) [lindex $coord 2]
    }
    $sel delete
}

foreach group [list [list $range1_start $range1_end] [list $range3_start $range3_end]] {
    lassign $group group_start group_end
    for {set ires $group_start} {$ires <= $group_end} {incr ires} {
        if {![info exists x($ires)]} { continue }

        for {set jres $range2_start} {$jres <= $range2_end} {incr jres} {
            if {![info exists x($jres)]} { continue }

            set dist [expr {
                sqrt(($x($ires)-$x($jres))**2 +
                     ($y($ires)-$y($jres))**2 +
                     ($z($ires)-$z($jres))**2)
            }]
            if {$dist < $crit} {
                set yellow_connected($jres) 1
            }
        }
    }
}

array unset adjacency_list
for {set ires $range2_start} {$ires <= $range2_end} {incr ires} {
    if {![info exists x($ires)]} { continue }

    for {set jres [expr {$ires + 1}]} {$jres <= $range2_end} {incr jres} {
        if {![info exists x($jres)]} { continue }

        set dist [expr {
            sqrt(($x($ires)-$x($jres))**2 +
                 ($y($ires)-$y($jres))**2 +
                 ($z($ires)-$z($jres))**2)
        }]
        if {$dist < $crit} {
            set red_connected($ires) 1
            set red_connected($jres) 1

            lappend adjacency_list($ires) $jres
            lappend adjacency_list($jres) $ires
        }
    }
}

set cluster_id 0
set total_cluster_size 0
set valid_cluster_count 0

for {set ires $range2_start} {$ires <= $range2_end} {incr ires} {
    if {![info exists x($ires)]} { continue }
    if {[info exists cluster_assigned($ires)]} { continue }

    set queue [list $ires]
    set current_cluster [list]
    set cluster_assigned($ires) 1

    while {[llength $queue] > 0} {
        set current [lindex $queue 0]
        set queue [lrange $queue 1 end]
        lappend current_cluster $current

        if {[info exists adjacency_list($current)]} {
            foreach neighbor $adjacency_list($current) {
                if {![info exists cluster_assigned($neighbor)]} {
                    set cluster_assigned($neighbor) 1
                    lappend queue $neighbor
                }
            }
        }
    }

    set size [llength $current_cluster]
```

```
        set cluster_size($cluster_id) $size

        if {$size >= $min_cluster_size} {
            incr valid_cluster_count
            incr total_cluster_size $size
        }

        incr cluster_id
    }

    if {$valid_cluster_count > 0} {
        set avg_cluster_size [expr {double($total_cluster_size) / $valid_cluster_count}]
    } else {
        set avg_cluster_size 0.0
    }

    set yellow_count [array size yellow_connected]
    set red_count [array size red_connected]

    puts $outfile [format "%d\t%d\t%d\t%d\t%.2f" $frame $yellow_count $red_count
$valid_cluster_count $avg_cluster_size]
    puts "Frame $frame: Yellow=$yellow_count, Red=$red_count, Clusters=$valid_cluster_count,
AvgSize=[format "%.2f" $avg_cluster_size]"
}

close $outfile
puts "Multi-frame statistics saved to $output_file"

animate goto [expr $num_frames - 1]
draw delete all

draw color blue
foreach {start end} [list $range1_start $range1_end $range3_start $range3_end] {
    for {set ires $start} {$ires <= $end} {incr ires} {
        if {[info exists x($ires)]} {
            draw sphere "$x($ires) $y($ires) $z($ires)" radius 1.5
        }
    }
}

draw color orange
```

```
for {set ires $range2_start} {$ires <= $range2_end} {incr ires} {
    if {[info exists x($ires)]} {
        draw sphere "$x($ires) $y($ires) $z($ires)" radius 1.5
    }
}

draw color yellow
foreach group [list [list $range1_start $range1_end] [list $range3_start $range3_end]] {
    lassign $group group_start group_end
    for {set ires $group_start} {$ires <= $group_end} {incr ires} {
        if {![info exists x($ires)]} { continue }

        for {set jres $range2_start} {$jres <= $range2_end} {incr jres} {
            if {![info exists x($jres)]} { continue }

            set dist [expr {
                sqrt(($x($ires)-$x($jres))**2 +
                     ($y($ires)-$y($jres))**2 +
                     ($z($ires)-$z($jres))**2)
            }]
            if {$dist < $crit} {
                draw cylinder "$x($ires) $y($ires) $z($ires)" \
                              "$x($jres) $y($jres) $z($jres)" \
                              radius 0.5
            }
        }
    }
}

draw color red
for {set ires $range2_start} {$ires <= $range2_end} {incr ires} {
    if {![info exists x($ires)]} { continue }

    for {set jres [expr {$ires + 1}]} {$jres <= $range2_end} {incr jres} {
        if {![info exists x($jres)]} { continue }

        set dist [expr {
            sqrt(($x($ires)-$x($jres))**2 +
                 ($y($ires)-$y($jres))**2 +
                 ($z($ires)-$z($jres))**2)
        }]
```

```
                if {$dist < $crit} {
                    draw cylinder "$x($ires) $y($ires) $z($ires)" \
                                "$x($jres) $y($jres) $z($jres)" \
                                radius 0.5
                }
        }
}
```