**Supplementary Information**

Measuring biomolecular DSC profiles with thermolabile ligands to rapidly characterize folding and binding interactions

Robert W. Harkness1, V, Philip E. Johnson2, and Anthony K. Mittermaier1

1Department of Chemistry, McGill University

801 Sherbrooke St. W.

Montreal Quebec, Canada

H3A 0B8

2Department of Chemistry, York University

4700 Keele St.

Toronto, Ontario, Canada

M3J 1P3

**Supplementary Methods**

*Characterizing non-equilibrium biomolecular folding and binding interactions with thermolabile ligands by DSC*

Non-equilibrium biomolecular folding and binding processes are identified by the presence of thermal hysteresis (a difference between the melting and annealing profile peak shapes and transition temperatures), or a lack of reproducibility of the scan signatures. These types of thermograms can be considered to be kinetically controlled and may be generally analyzed with rate equations describing the rates of change of concentrations for each of the species involved1,2. This approach can be extended to DSC binding experiments with thermolabile ligands by including an equation governing the ligand concentration as a function of time and temperature in the DSC experiment. Here we consider the model in Scheme 1 of the main text for a biomolecule undergoing thermolabile ligand binding, folding, and aggregation processes.

We simulated scenarios using the model in Scheme 1 where aggregation occurs or biomolecular folding and ligand binding are kinetically controlled. The simulation temperatures were defined according to

 (1)

where *T0* and *Tf*are the initial and final temperatures for each simulated DSC scan, *n* is the increment number, *Δtint* is the integration time increment (s) (see below), and *ttot* (s) is the total length of time for the DSC scan calculated by

 (2)

where  is the scan rate (°C s-1). The simulations were bounded by the total number of increments *ntot* computed with

 (3)

such that when *n* = *ntot,* *Tn* = *Tf*.

Considering Scheme 1 in the main text, the changes in each species concentration with respect to time are given by

 (4)

 (5)

 (6)

 (7)

 (8)

and the rate constants are given by

 (9)

where *A* is the pre-exponential factor, *Ea* is the activation enthalpy, and *R* is the ideal gas constant. The concentrations of each species can be computed numerically starting from an initial condition using equations of the form

 (10)

For brevity, we have demonstrated the integration with just the ligand concentration. We simulated DSC profiles with a scan rate of 1 °C min-1. The populations of each biomolecular state and their temperature derivatives are calculated according to

 (11)

 (12)

Where  is the inverse scan rate and is the total concentration of biomolecule. We have shown the calculation for just the bound state population for brevity. The excess heat capacity function for Scheme 1 is given by

 (13)

calculated relative to the reference folded state. Here we have chosen temperature-independent changes in enthalpy relative to the folded state for simplicity. If required, the temperature dependences of  are accounted for by including the  parameter. The changes in enthalpy for the reversible folding and binding steps were calculated according to

 (14)

and

 (15)

with simulation values of 200 and -140 kJ mol-1 respectively. The change in enthalpy for aggregation relative to the folded state was chosen as 50 kJ mol-1. Arrhenius parameters for equilibrium binding and folding were Aon = 5x10-1 M-1 s-1, Aoff = 1x1019 s-1, Eaon = -20 kJ mol-1, Eaoff = 120 kJ mol-1, Afold= 1x10-14 s-1, Aunfold = 5x1018, Eafold = -80 kJ mol-1, and Eaunfold = 120 kJ mol-1. Arrhenius parameters for kinetically-controlled binding and folding were Aon = 5x10-3 M-1 s-1, Aoff = 1x1016 s-1, Eaon = -20 kJ mol-1, Eaoff = 120 kJ mol-1, Afold= 1x10-16 s-1, Aunfold = 5x1016, Eafold = -80 kJ mol-1, and Eaunfold = 120 kJ mol-1. Arrhenius parameters for slow and rapid thermolabile ligand conversion were Aslow= 7.509x1010 s-1, Easlow = 94.65 kJ mol-1, and Afast= 1 s-1, Eafast = 10 kJ mol-1. Arrhenius parameters for slow irreversible aggregation were Aagg.=5x107 s-1 and Eaagg.= 80 kJ mol-1. The simulations were performed with lower and upper temperatures of 0 and 80 °C and a scan rate of 1 °C min-1. 20 scans (10 melting and 10 annealing) were simulated with total biomolecule and ligand concentrations of 200 µM and 10 mM respectively. The concentrations of all species were allowed to equilibrate at 0 °C for ten minutes to simulate the pre-scan equilibration time in the calorimeter. After each scan, the concentrations of each species were allowed to equilibrate for 60 s. The MATLAB code for generating kinetically controlled DSC datasets with thermolabile ligands (including the requisite parameters) is given below.

## **Simulating kinetically controlled DSC experiments with thermolabile ligands.**

function simulatekinDSC

## 1. Model is B 🡨🡪 F 🡨🡪 L 🡨🡪 U + L 🡪 A + L

d[B]/dt = kon[F][L] - koff[B]

d[F]/dt = koff[B] + kfold[U] - kon[F][L] - kunfold[F]

d[U]/dt = kunfold[F] - kfold[U] - kaggregate[U]

d[A]/dt = kaggregate[U]

d[L]/dt = koff[B] - kon[F][L] - kconversion[L]

## 2. Initialize simulation parameters

Set up scan rate, number of scans time integration steps, boundary temperatures

% Convert dTdt to 1/s!  
dTdt = [1];  
dTdt = dTdt./60;  
dtdT = 1./dTdt;  
  
% Choose deltat such that the shapes of simulated curves do not depend on  
% this value and the simulated concentrations do not reach NaN  
deltat = 1e-3;  
deltaT = deltat./dtdT;  
  
% Tspace is for allowing To and Tf to swap after completion of each scan  
To = 0;  
Tf = 80;  
Tspace = To;  
  
nscans = 20;  
nsteps = (Tf - To)./deltaT  
ttot = (Tf - To)./(dTdt);  
  
% Choose how often to sample the computed concentrations so as to not  
% overwhelm the computer's memory  
storeinc = 1e3;

## 

## 3. Set up kinetic and thermodynamic parameters in kJ mol-1, define biomolecule and ligand concentrations

R = 8.3145e-3;  
  
% Fast binding parameters  
Aon = 5e-1;  
Eaon = -20;  
Aoff = 1e19;  
Eaoff = 120;  
  
% Slow binding parameters  
% Aon = 5e-3;  
% Eaon = -20;  
% Aoff = 1e16;  
% Eaoff = 120;  
  
% Fast folding and unfolding parameters  
Afold = 1e-14;  
Eafold = -80;  
Aunfold = 5e18;  
Eaunfold = 120;  
  
% Slow folding and unfolding parameters  
% Afold = 1e-16;  
% Eafold = -80;  
% Aunfold = 5e16;  
% Eaunfold = 120;  
  
% Slow highly temperature-dependent thermolabile ligand conversion parameters  
Aconversion = 7.509e10;  
Eaconversion = 94.65;  
  
% Ultra slow highly temperature-dependent thermolabile ligand conversion parameters  
% Aconversion = 7.509e-20;  
% Eaconversion = 94.65;  
  
% Fast weakly temperature-dependent thermolabile ligand conversion  
% Aconversion = 1e0;  
% Eaconversion = 10;  
  
% Slow irreversible aggregation of the biomolecule parameters  
Aaggregate = 5e7;  
Eaaggregate = 80;  
  
% For turning off steps in the model  
% Aon = 0;  
% Aoff = 0;  
% Afold = 0;  
% Aunfold = 0;  
% Aconversion = 0;  
% Aaggregate = 0;  
  
dHBF = Eaon - Eaoff;  
dHUF = Eaunfold - Eafold;  
dHAF = 50;  
  
CT = 200e-6;  
LT = 10000e-6;  
  
% Plot rate constants to check that they are calculated as desired  
T = [0:0.1:80]' + 273;  
  
kon = Aon.\*exp(-Eaon./(R.\*T));  
koff = Aoff.\*exp(-Eaoff./(R.\*T));  
kfold = Afold.\*exp(-Eafold./(R.\*T));  
kunfold = Aunfold.\*exp(-Eaunfold./(R.\*T));  
kaggregate = Aaggregate.\*exp(-Eaaggregate./(R.\*T));  
kconversion = Aconversion.\*exp(-Eaconversion./(R.\*T));  
  
figure  
plot(T,log10(kfold),'b','LineWidth',2)  
hold on  
plot(T,log10(kunfold),'r','LineWidth',2)  
plot(T,log10(kon),'g','LineWidth',2)  
plot(T,log10(koff),'m','LineWidth',2)  
plot(T,log10(kaggregate),'c','LineWidth',2)  
plot(T,log10(kconversion),'y','LineWidth',2)  
hold off  
set(gcf,'color','w')  
set(gca,'LineWidth',2,'FontSize',12)  
xlabel(['Temperature K'])  
ylabel('log\_{10}(k)')  
legend('kfold', 'kunfold', 'kon', 'koff', 'kagg', 'kconv')  
hold off  
  
clear T kfold kunfold kon koff kaggregate kconversion

## 

## 4. Simulate experiment, store populations and dP/dTs according to store increment

for ii = 1:size(dTdt,1)  
  
 ii  
  
 % Let concentrations reach equilibrium before starting simulation  
  
 preteq = 600;  
 neqsteps = preteq./deltat;  
  
 % Choose initial conditions for the type of desired experiment  
 Beqt = CT;  
 %Beqt = 0;  
  
 Feqt = 0;  
 %Feqt = CT;  
  
 Ueqt = 0;  
  
 Aeqt = 0;  
  
 Leqt = LT - CT;  
 %Leqt = 0;  
  
 T = To + 273;  
  
 kon = Aon.\*exp(-Eaon./(R.\*T));  
 koff = Aoff.\*exp(-Eaoff./(R.\*T));  
 kfold = Afold.\*exp(-Eafold./(R.\*T));  
 kunfold = Aunfold.\*exp(-Eaunfold./(R.\*T));  
 kaggregate = Aaggregate.\*exp(-Eaaggregate./(R.\*T));  
 kconversion = Aconversion.\*exp(-Eaconversion./(R.\*T));  
  
 for nt = 1:neqsteps  
  
 Beqtplus1 = Beqt + (kon.\*Feqt.\*Leqt - koff.\*Beqt).\*deltat;  
  
 Feqtplus1 = Feqt + (koff.\*Beqt + kfold.\*Ueqt - kon.\*Feqt.\*Leqt - kunfold.\*Feqt).\*deltat;  
  
 Ueqtplus1 = Ueqt + (kunfold.\*Feqt - kfold.\*Ueqt - kaggregate.\*Ueqt).\*deltat;  
  
 Aeqtplus1 = Aeqt + (kaggregate.\*Ueqt).\*deltat;  
  
 Leqtplus1 = Leqt + (koff.\*Beqt - kon.\*Feqt.\*Leqt - kconversion.\*Leqt).\*deltat;  
  
 Beqt = Beqtplus1;  
  
 Feqt = Feqtplus1;  
  
 Ueqt = Ueqtplus1;  
  
 Aeqt = Aeqtplus1;  
  
 Leqt = Leqtplus1;  
  
 end  
  
 Bt = Beqt;  
  
 Ft = Feqt;  
  
 Ut = Ueqt;  
  
 At = Aeqt;  
  
 Lt = Leqt;  
  
 incs = 1;  
  
 for scans = 1:nscans  
  
 scans  
  
 Tstore = zeros(nsteps(ii,1)./storeinc,1);  
  
 PBstore = zeros(nsteps(ii,1)./storeinc,1);  
 PFstore = zeros(nsteps(ii,1)./storeinc,1);  
 PUstore = zeros(nsteps(ii,1)./storeinc,1);  
 PAstore = zeros(nsteps(ii,1)./storeinc,1);  
 PLstore = zeros(nsteps(ii,1)./storeinc,1);  
  
 dPBdTstore = zeros(nsteps(ii,1)./storeinc,1);  
 dPFdTstore = zeros(nsteps(ii,1)./storeinc,1);  
 dPUdTstore = zeros(nsteps(ii,1)./storeinc,1);  
 dPAdTstore = zeros(nsteps(ii,1)./storeinc,1);  
  
 for nt = 1:nsteps(ii,1)./storeinc  
  
 for ntstore = 1:storeinc  
  
 T = To + (Tf - To).\*((incs.\*deltat)./ttot(ii,1));  
 T = T + 273;  
  
 kon = Aon.\*exp(-Eaon./(R.\*T));  
 koff = Aoff.\*exp(-Eaoff./(R.\*T));  
 kfold = Afold.\*exp(-Eafold./(R.\*T));  
 kunfold = Aunfold.\*exp(-Eaunfold./(R.\*T));  
 kaggregate = Aaggregate.\*exp(-Eaaggregate./(R.\*T));  
 kconversion = Aconversion.\*exp(-Eaconversion./(R.\*T));  
  
 Btplus1 = Bt + (kon.\*Ft.\*Lt - koff.\*Bt).\*deltat;  
 dPBdT = (dtdT(ii,1).\*(kon.\*Ft.\*Lt - koff.\*Bt))./CT;  
  
 Ftplus1 = Ft + (koff.\*Bt + kfold.\*Ut - kon.\*Ft.\*Lt - kunfold.\*Ft).\*deltat;  
 dPFdT = (dtdT(ii,1).\*(koff.\*Bt + kfold.\*Ut - kon.\*Ft.\*Lt - kunfold.\*Ft))./CT;  
  
 Utplus1 = Ut + (kunfold.\*Ft - kfold.\*Ut - kaggregate.\*Ut).\*deltat;  
 dPUdT = (dtdT(ii,1).\*(kunfold.\*Ft - kfold.\*Ut - kaggregate.\*Ut))./CT;  
  
 Atplus1 = At + (kaggregate.\*Ut).\*deltat;  
 dPAdT = (dtdT(ii,1).\*(kaggregate.\*Ut))./CT;  
  
 Ltplus1 = Lt + (koff.\*Bt - kon.\*Ft.\*Lt - kconversion.\*Lt).\*deltat;  
  
 Bt = Btplus1;  
  
 Ft = Ftplus1;  
  
 Ut = Utplus1;  
  
 At = Atplus1;  
  
 Lt = Ltplus1;  
  
 incs = incs + 1;  
  
 end  
  
 Tstore(nt,1) = T;  
  
 PBstore(nt,1) = Bt./CT;  
 PFstore(nt,1) = Ft./CT;  
 PUstore(nt,1) = Ut./CT;  
 PAstore(nt,1) = At./CT;  
 PLstore(nt,1) = Lt./LT;  
  
 dPBdTstore(nt,1) = dPBdT;  
 dPFdTstore(nt,1) = dPFdT;  
 dPUdTstore(nt,1) = dPUdT;  
 dPAdTstore(nt,1) = dPAdT;  
  
 end  
  
 Texp{1,ii}{:,scans} = Tstore;  
  
 PBexp{1,ii}{:,scans} = PBstore;  
 PFexp{1,ii}{:,scans} = PFstore;  
 PUexp{1,ii}{:,scans} = PUstore;  
 PAexp{1,ii}{:,scans} = PAstore;  
 PLexp{1,ii}{:,scans} = PLstore;  
  
 dPBdTexp{1,ii}{:,scans} = dPBdTstore;  
 dPFdTexp{1,ii}{:,scans} = dPFdTstore;  
 dPUdTexp{1,ii}{:,scans} = dPUdTstore;  
 dPAdTexp{1,ii}{:,scans} = dPAdTstore;  
  
 clear Tstore PBstore PFstore PUstore PAstore PLstore dPBdTstore dPFdTstore dPUdTstore dPAdTstore  
  
 % Equilibrate at the end of each scan  
  
 teqscan = 60;  
 nscaneqsteps = teqscan./deltat;  
  
 T = To + (Tf - To).\*((incs.\*deltat)./ttot(ii,1));  
 T = T + 273;  
  
 Beqt = Bt;  
  
 Feqt = Ft;  
  
 Ueqt = Ut;  
  
 Aeqt = At;  
  
 Leqt = Lt;  
  
 kon = Aon.\*exp(-Eaon./(R.\*T));  
 koff = Aoff.\*exp(-Eaoff./(R.\*T));  
 kfold = Afold.\*exp(-Eafold./(R.\*T));  
 kunfold = Aunfold.\*exp(-Eaunfold./(R.\*T));  
 kaggregate = Aaggregate.\*exp(-Eaaggregate./(R.\*T));  
 kconversion = Aconversion.\*exp(-Eaconversion./(R.\*T));  
  
 for nt = 1:nscaneqsteps  
  
 Beqtplus1 = Beqt + (kon.\*Feqt.\*Leqt - koff.\*Beqt).\*deltat;  
  
 Feqtplus1 = Feqt + (koff.\*Beqt + kfold.\*Ueqt - kon.\*Feqt.\*Leqt - kunfold.\*Feqt).\*deltat;  
  
 Ueqtplus1 = Ueqt + (kunfold.\*Feqt - kfold.\*Ueqt - kaggregate.\*Ueqt).\*deltat;  
  
 Aeqtplus1 = Aeqt + (kaggregate.\*Ueqt).\*deltat;  
  
 Leqtplus1 = Leqt + (koff.\*Beqt - kon.\*Feqt.\*Leqt - kconversion.\*Leqt).\*deltat;  
  
 Beqt = Beqtplus1;  
  
 Feqt = Feqtplus1;  
  
 Ueqt = Ueqtplus1;  
  
 Aeqt = Aeqtplus1;  
  
 Leqt = Leqtplus1;  
  
 end  
  
 % Next scan concentrations become equilibrated concentrations,  
 % reverse To and Tf and scan again  
  
 Bt = Beqt;  
  
 Ft = Feqt;  
  
 Ut = Ueqt;  
  
 At = Aeqt;  
  
 Lt = Leqt;  
  
 incs = 1;  
  
 To = Tf;  
 Tf = Tspace;  
 Tspace = To;  
  
 end  
  
end

## 

## 5. Calculate excess heat capacity and plot it along with the populations of each state, save results of the simulation

for ii = 1:size(dTdt,1)  
  
 for kk = 1:nscans  
  
 Cpcalc{1,ii}{:,kk} = dPUdTexp{1,ii}{:,kk}.\*dHUF + dPBdTexp{1,ii}{:,kk}.\*dHBF + dPAdTexp{1,ii}{:,kk}.\*dHAF;  
  
 end  
  
end  
  
figure  
cmap = lbmap(nscans,'RedBlue');  
  
for ii = 1:size(dTdt,1)  
  
 for kk = 1:nscans  
  
 plot(Texp{1,ii}{:,kk}-273,Cpcalc{1,ii}{:,kk},'color',cmap(kk,:),'LineWidth',2)  
 hold on  
  
 end  
  
end  
  
set(gcf,'color','w')  
set(gca,'LineWidth',2,'FontSize',12,'XColor','k','YColor','k')  
xlabel(['Temperature ', char(176), 'C'])  
ylabel('Excess C\_{p} kJ mol^{-1} K^{-1}')  
hold off  
  
figure  
  
for ii = 1:size(dTdt,1)  
  
 for kk = 1:nscans  
  
 plot(Texp{1,ii}{:,kk}-273,PFexp{1,ii}{:,kk},'b','LineWidth',2)  
 hold on  
 plot(Texp{1,ii}{:,kk}-273,PUexp{1,ii}{:,kk},'r','LineWidth',2)  
 plot(Texp{1,ii}{:,kk}-273,PBexp{1,ii}{:,kk},'g','LineWidth',2)  
 plot(Texp{1,ii}{:,kk}-273,PAexp{1,ii}{:,kk},'c','LineWidth',2)  
 plot(Texp{1,ii}{:,kk}-273,PLexp{1,ii}{:,kk},'y','LineWidth',2)  
  
 end  
  
end  
  
set(gcf,'color','w')  
set(gca,'LineWidth',2,'FontSize',12)  
xlabel(['Temperature ', char(176), 'C'])  
ylabel('Population')  
legend('PF', 'PU', 'PB', 'PA', 'PL')  
hold off  
  
save('kinDSC.mat', 'Texp', 'PBexp', 'PFexp', 'PUexp', 'PAexp', 'PLexp', 'dPBdTexp', 'dPFdTexp', 'dPUdTexp', 'dPAdTexp', 'Cpcalc')

end

**References**

1 Prislan, I., Lah, J. & Vesnaver, G. Diverse polymorphism of G-quadruplexes as a kinetic phenomenon. *J. Am. Chem. Soc.* **130** (43), 14161-14169, doi:10.1021/ja8026604, (2008).

2 Toledo-Nunez, C., Vera-Robles, L. I., Arroyo-Maya, I. J. & Hernandez-Arana, A. Deconvolution of complex differential scanning calorimetry profiles for protein transitions under kinetic control. *Anal. Biochem.* **509** 104-110, doi:10.1016/j.ab.2016.07.006, (2016).