

```

1 function [fn]=updatclim_coawst_mw(T1, gn, clm, clmname, wdr, url)
2 % Modified by Brandy Armstrong January 2012 to use only NCTOOLBOX
3 % and Matlab builtin functions to read and write netcdf files
4 % jcw Feb 2019 - only use matalb BI
5 %
6 %T1 = date for climatology file
7 %gn = data from grid
8 %clm = data of hycom indices
9 %wdr = the working directory
10 %clmname = grid name prefix for climatology filenames
11 %url = where get data from
12 %
13 %
14 %determine indices for time period of interpolation
15 %
16 disp('getting the number of time records ...');
17 % jsasaki
18 % tr0=datenum(0,1,1); % => 1 (day)
19 % disp(tr0);
20 t0=datenum(1900,12,31); % tr0=datenum(1858,11,17);
21 disp(t0);
22 % jsasaki
23 %time=ncread(url,'MT');
24 time=ncread(url,'time');
25 disp(time(1));
26 disp(time(2));
27 tg=time+t0;
28 tg2=julian(str2num(datestr(tg,'yyyy')),str2num(datestr(tg,'mm')),str2num(datestr(tg,'dd')),str2num(datestr(tg,'HH')))-2400001;
29 %
30 % get user times
31 %
32 [junk,tid1,ib]=intersect(tg,floor(T1));%modify to be nearest jcw 23Aug2014
33 if isempty(tid1)
34     tid1=length(tg);
35 end
36
37 fn=[clmname];
38 disp(['creating netcdf file ',fn]);
39 create_roms_netcdf_clm_mwUL(fn,gn,1);% converted to BI functions
40
41 %fill grid dims using builtin (BI) functions
42 RN=netcdf.open(fn,'NC_WRITE');
43 lonid=netcdf.inqVarID(RN,'lon_rho');
44 netcdf.putVar(RN,lonid,gn.lon_rho);
45 latid=netcdf.inqVarID(RN,'lat_rho');
46 netcdf.putVar(RN,latid,gn.lat_rho);
47 netcdf.close(RN)
48
49 %%
50 tz_levs=length(clm.z);
51 X= repmat(clm.lon,1,length(clm.lat));
52 Y= repmat(clm.lat,length(clm.lon),1);
53 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
54 disp(['Interpolating u for ',datestr(tg(tid1))]);
55 ttu=1;
56 clm.u=zeros([length(clm.z) size(gn.lon_rho)]);
57 while ttu==1;
58     try
59         tmpt=ncread(url,'u',[clm.ig0 clm.jg0 1 tid1],[clm.ig1-clm.ig0+1 clm.jg1-clm.jg0+1 tz_l
60             for k=1:tz_levs
61                 disp(['doing griddata u for HYCOM level ' num2str(k)]);
62                 tmp=double(squeeze(tmpt(:, :, k)));
63                 F = scatteredInterpolant(X(:),Y(:),tmp(:));
64                 cff = F(gn.lon_rho,gn.lat_rho);
65                 clm.u(k, :, :)=maplev(cff);
66             end
67         ttu=0;

```

time(1)=105204

time(end)=113985

2000/1/1 からの h

時点に3リッル日付値に統一する。

HYCOM は 2000-1-1 00:00:00

を基準と可子 hours などで、

これを3リッル日付値に変更する。

修正ユリウス日の元期

その後、修正ユリウス日に変換

現在の暦 ユリウス通日

mtods/julian.m Gregorian → Julian dates

3リッル日付値  
修正ユリウス日

修正ユリウス日 = ユリウス暦紀元前4713年1月1日正子(UTC) 正子=午前0時

修正ユリウス日 0 = グレゴリ暦 1858年11月17日正子(UTC)

lon lat depth time

始点 (1225, 1850, 1, 2883)

個数 (215, 228, 40, 1)

start loc

count

2

1225 1850 2883 215 228 40

```

68     catch
69         disp(['catch u Unable to download HYCOM u data at' datestr(now)]);
70         fid=fopen('coawstlog.txt','a');
71         fprintf(fid,'Unable to download HYCOM u data at');
72         fprintf(fid,datestr(now));
73         fprintf(fid,'\n');
74     end
75 end
76 %== Vertical interpolation (t,s,u,v) from standard z-level to s-level
77 u=roms_from_stdlev_mw(gn.lon_rho,gn.lat_rho,clm.z,clm.u,gn,'u',0);
78 clm=rmfield(clm,'u');
79 save u.mat u
80 clear u;
81
82 disp(['Interpolating v for ',datestr(tg(tid1))]);
83 ttv=1;
84 clm.v=zeros([length(clm.z) size(gn.lon_rho)]);
85 while ttv==1;
86     try
87         tmp=ncread(url,'v',[clm.ig0 clm.jg0 1 tid1],[clm.ig1-clm.ig0+1 clm.jg1-clm.jg0+1 tz_1
88         evs 1 ] );
89         for k=1:tz_levs
90             disp(['doing griddata v for HYCOM level ' num2str(k)]);
91             tmp=double(squeeze(tmp(:,:,k)));
92             F = scatteredInterpolant(X(:),Y(:),tmp(:));
93             cff = F(gn.lon_rho,gn.lat_rho);
94             clm.v(k,:,:) = maplev(cff);
95         end
96         ttv=0;
97     catch
98         disp(['catch v Unable to download HYCOM v data at' datestr(now)]);
99         fid=fopen('coawstlog.txt','a');
100        fprintf(fid,'Unable to download HYCOM v data at');
101        fprintf(fid,datestr(now));
102        fprintf(fid,'\n');
103    end
104 %== Vertical interpolation (t,s,u,v) from standard z-level to s-level
105 v=roms_from_stdlev_mw(gn.lon_rho,gn.lat_rho,clm.z,clm.v,gn,'v',0);
106 clm=rmfield(clm,'v');
107 save v.mat v
108 clear v;
109
110 %== Rotate the velocity
111 theta=exp(-sqrt(-1)*mean(mean(gn.angle)));
112 load u.mat; load v.mat
113 disp('doing rotation to grid for u and v');
114 uv=(u2rho_3d_mw(u)+sqrt(-1)*v2rho_3d_mw(v)).*theta;
115 u=rho2u_3d_mw(real(uv)); v=rho2v_3d_mw(imag(uv));
116 clear uv
117
118 %% == output
119 RN=netcdf.open(fn,'NC_WRITE');
120
121 tempid=netcdf.inqVarID(RN,'u');
122 netcdf.putVar(RN,tempid,shiftdim(u,1));
123
124 tempid=netcdf.inqVarID(RN,'v');
125 netcdf.putVar(RN,tempid,shiftdim(v,1));
126
127 clear u; clear v;
128 tempid=netcdf.inqVarID(RN,'ocean_time');
129 netcdf.putVar(RN,tempid,tg2(tid1));
130 tempid=netcdf.inqVarID(RN,'zeta_time');
131 netcdf.putVar(RN,tempid,tg2(tid1));
132 tempid=netcdf.inqVarID(RN,'v2d_time');
133 netcdf.putVar(RN,tempid,tg2(tid1));
134 tempid=netcdf.inqVarID(RN,'v3d_time');
135 netcdf.putVar(RN,tempid,tg2(tid1));

```

OPeNDAPの ncread() はしばしば  
失敗するので、try ~ catch で  
成功するまで実行する。

失敗すると ttv = 1 のままなので、成功するまでくり返す。  
失敗するのを前提に設計されている。

ncread() は常に  
try ~ catch で実行する。

```

136 tempid=netcdf.inqVarID(RN,'salt_time');
137 netcdf.putVar(RN,tempid,tg2(tid1));
138 tempid=netcdf.inqVarID(RN,'temp_time');
139 netcdf.putVar(RN,tempid,tg2(tid1));
140 netcdf.close(RN);
141 %%
142 %== Depth averaging u, v to get Ubar
143 load u.mat; load v.mat
144 cc=roms_zint_mw(u,gn); ubar=rho2u_2d_mw(u2rho_2d_mw(cc)./gn.h);
145 cc=roms_zint_mw(v,gn); vbar=rho2v_2d_mw(v2rho_2d_mw(cc)./gn.h);
146 %== Rotate the velocity
147 uv=(u2rho_2d_mw(ubar)+sqrt(-1)*v2rho_2d_mw(vbar)).*theta;
148 ubar=rho2u_2d_mw(real(uv)); vbar=rho2v_2d_mw(imag(uv));
149 clear u
150 clear v
151
152 RN=netcdf.open(fn,'NC_WRITE');
153 tempid=netcdf.inqVarID(RN,'ubar');
154 netcdf.putVar(RN,tempid,ubar);
155 tempid=netcdf.inqVarID(RN,'vbar');
156 netcdf.putVar(RN,tempid,vbar);
157 netcdf.close(RN);
158
159 clear ubar
160 clear vbar
161 clear uv
162 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
163 %% interpolate the zeta data
164 disp(['Interpolating zeta for ',datestr(tg(tid1))]);
165 ttz=1;
166 while ttz==1;
167     try
168         tmp=ncread(url,'ssh',[clm.ig0 clm.jg0 tid1],[clm.ig1-clm.ig0+1 clm.jg1-clm.jg0+1 1 ]
169 );
170         tmp=double(squeeze(tmp(:,:,)));
171         disp(['doing griddata zeta for HYCOM ']);
172         F = scatteredInterpolant(X(:),Y(:),tmp(:));
173         cff = F(gn.lon_rho,gn.lat_rho);
174         zeta=maplev(cff);
175         ttz=0;
176     catch
177         disp(['catch z Unable to download HYCOM ssh data at' datestr(now)]);
178         fid=fopen('coawstlog.txt','a');
179         fprintf(fid,'Unable to download HYCOM ssh data at');
180         fprintf(fid,datestr(now));
181         fprintf(fid,'\n');
182     end
183 end
184 clear tmp
185 %
186 %== output zeta
187 %
188 RN=netcdf.open(fn,'NC_WRITE');
189 tempid=netcdf.inqVarID(RN,'zeta');
190 netcdf.putVar(RN,tempid,zeta);
191 netcdf.close(RN);
192 clear zeta;
193 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
194 disp(['Interpolating temp for ',datestr(tg(tid1))]);
195 ttt=1;
196 clm.temp=zeros([length(clm.z) size(gn.lon_rho)]);
197 while ttt==1;
198     try
199         tmp=ncread(url,'temperature',[clm.ig0 clm.jg0 1 tid1],[clm.ig1-clm.ig0+1 clm.jg1-clm.
200 jg0+1 tz_levs 1 ] );
201         for k=1:tz_levs
202             disp(['doing griddata temp for HYCOM level ' num2str(k)]);
203             tmp=double(squeeze(tmp(:,:,k)));
204             F = scatteredInterpolant(X(:),Y(:),tmp(:));

```

```

203         cff = F(gn.lon_rho,gn.lat_rho);
204 %         cff(cff<0)=nan;
205         clm.temp(k,:,:) = maplev(cff);
206     end
207     ttt=0;
208     catch
209         disp(['catch temp Unable to download HYCOM temp data at' datestr(now)]);
210         fid=fopen('coawstlog.txt','a');
211         fprintf(fid,'Unable to download HYCOM temp data at');
212         fprintf(fid,datestr(now));
213         fprintf(fid,'\n');
214     end
215 end
216 %
217 %== Vertical interpolation (t,s,u,v) from standard z-level to s-level
218 %
219 temp=roms_from_stdlev_mw(gn.lon_rho,gn.lat_rho,clm.z,clm.temp,gn,'rho',0);
220 clm=rmfield(clm,'temp');
221 %
222 %== output temp
223 %
224 RN=netcdf.open(fn,'NC_WRITE');
225 tempid=netcdf.inqVarID(RN,'temp');
226 netcdf.putVar(RN,tempid,shiftdim(temp,1));
227 netcdf.close(RN);
228 clear temp;
229
230
231 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
232 disp(['Interpolating salt for ',datestr(tg(tid1))]);
233 tts=1;
234 clm.salt=zeros([length(clm.z) size(gn.lon_rho)]);
235 while tts==1;
236     try
237         tmp1=ncread(url,'salinity',[clm.ig0 clm.jg0 1 tid1],[clm.ig1-clm.ig0+1 clm.jg1-clm.jg0
+1 tz_levs 1 ] );
238         for k=1:tz_levs
239             disp(['doing griddata salt for HYCOM level ' num2str(k)]);
240             tmp=double(squeeze(tmp1(:,:,k)));
241             F = scatteredInterpolant(X(:),Y(:),tmp(:));
242             cff = F(gn.lon_rho,gn.lat_rho);
243             cff(cff<0)=nan;
244             clm.salt(k,:,:) = maplev(cff);
245         end
246         tts=0;
247     catch
248         disp(['catch temp Unable to download HYCOM temp data at' datestr(now)]);
249         fid=fopen('coawstlog.txt','a');
250         fprintf(fid,'Unable to download HYCOM temp data at');
251         fprintf(fid,datestr(now));
252         fprintf(fid,'\n');
253     end
254 end
255 %
256 %== Vertical interpolation (t,s,u,v) from standard z-level to s-level
257 %
258 salt=roms_from_stdlev_mw(gn.lon_rho,gn.lat_rho,clm.z,clm.salt,gn,'rho',0);
259 clm=rmfield(clm,'salt');
260 %
261 %== output salt
262 %
263 RN=netcdf.open(fn,'NC_WRITE');
264 tempid=netcdf.inqVarID(RN,'salt');
265 netcdf.putVar(RN,tempid,shiftdim(salt,1));
266 netcdf.close(RN);
267 clear salt;
268
269 disp(['Finished creating clim file at ' datestr(now)]);
270 %%

```