

Thermal model for generation of thermal movies

Hugh H. Kieffer File= /xtex/tes/krc/mkrc.tex

October 24, 2011

Contents

1	Introduction	2
2	Design considerations	2
2.1	Approach	2
2.2	Some details	2
3	Source maps and pre-processing; maprebin.pro	3
4	Thermal model calculation	4
4.1	new code	4
4.1.1	Main program; mkrc.f	4
4.1.2	Albedo, inertia and elevation for a zone: readaie.f.	5
4.2	mkrc Input file	5
5	Tiling: krctile.pro	6
5.0.1	Some size options	6
6	Timing results	6
6.0.2	mkrc	6
6.0.3	KRCtile	7
A	Some of the email exchanges with Phil	7
A.0.4	08/02/2011 03:36 PM	7
A.0.5	08/17/2011 09:32 AM	8
A.0.6	08/18/2011 05:29 AM	8
A.0.7	08/18/2011 05:55 AM	8
A.0.8	08/18/2011 06:11 AM	8
A.0.9	08/20/2011 04:39 PM	8
B	IDL-FORTRAN bug	8

1 Introduction

This paper describes a numerical model designed to generate surface temperatures for a movie that shows global diurnal temperatures. It was generated in August 13 to 22, 2011 in hopes of speeding up the generation of data for a movie for a single season, which took 2 days on a dedicated 128 cpu cluster when running KRC in the one-point mode. My initial “off-the-cuff” estimate was factor of 100 to 1000 faster. Achieved speedup is factor of about 3800.

1.0.1 History

History of this document:

2011sep17 First draft

2011oct17 Version sent to ASU

2011oct24 Minor typo and cleanup

2 Design considerations

1. Must support map resolution of $1/8^\circ$, and corresponding time resolution.
2. Allow at least Albedo, Thermal inertia and elevation to be specified to each point on the map.
3. Use the KRC thermal algorithms
4. Output in a format that will be compatible with Phil’s movie-generation scheme.
5. Execute in much less than 256 cpu days.

There is the potential for some really large files: a global simple-cylindrical map of $1/8^\circ$ resolution and corresponding hour resolution involves $180 \cdot 360^2 \cdot 8^3 = 1.2 \times 10^{10}$ words. Because of this, consider 8- and 16-bit words where practical.

2.1 Approach

1. Compute all times of day in a tight loop. This will be the major speed improvement.
2. Put the longitude loop inside the latitude loop so do not have to recalculate the insolation geometry. However, must do Delta-Eddington calculations for each local opacity, which is dependant upon elevation.
3. Minimize the number of KRC routines that are changed.
4. Retain the normal KRC input parameter format where possible.
5. Make a single file containing, albedo, thermal inertia and elevation with movie resolution, all with the proper lat/long. conventions.
Could make this file direct access if it were generated in FORTRAN
6. Use IDL where practical.
7. Use intermediate files to simplify software complexity.
8. Use BIN5 files where practical.
9. Because goal is a single (but any) season, solar geometry is fixed for an entire run.
Input DAU and SDEC, rather than use the PORB system.

2.2 Some details

MAXN2 and MAXNH in KRCCOM limit the number of output hours. Both are expected to be multiples of 24. (for Hour output) and 64 (for up to 6 time doublings). However, 1440 is a multiple of 24 and 32.

Normal KRC limit on MAXN2 is 768, corresponding to less than 2 minutes delta t, or less than 1/2 degree surface slope. Decide that linear interpolation to finer time resolution will be adequate.

KRC file output is 768 hours * 360*8 longitudes * 8 latitudes = 17.7M words per 1° strip. Use of scaled temperature allows 2-bit words, or 35.4 Mbyte files.

Output temperature as (T-220.)* 100 as I2 words. Considering Mars' range of temperatures is within 120-320K, could even go to *320 without overflow.

Pick an Ls, compute the date

Run for N3 days with output at 738 times

----- Code changes -----

KRCCOM: adjust sizes for optimum

KRC:

Loop on latitude, Call version of TLATS directly, Generate output file name
Write binary output calling BINf5 directly

TLATS > MLATS: Loop only on pixel, not lat. A single call will be a single lat!
Transfer new items via arguments rather than modifying commons.

Does IQ=arq1 pixels per lat strip <= MAXN4

Plan: Work toward tiles of <1GB, prefer division in longitude, as only half the planet is visible in one frame. BUT, if sub-view point is at latitude that exceeds width of the polar band, still need all longitudes for a frame.

MKRC writes 1-degree files of 16-bit T, 768 times, each is 35.3 MB
[Files are named to nearest 0.1 degree]

Tiling: Loop on Lat bands, write all longitude tiles for that band.

krctile.pro has a section for computing files sizes and the amount of memory needed, action @40. See §5.0.1.

Decide to output KRC in files corresponding to one degree of latitude.

For polar latitudes, could create a file of all zeros, or value corresponding to some reasonable polar temperature, and create soft links to it for all polar 1-degree bands.

3 Source maps and pre-processing; maprebin.pro

Note, program **maprebin** contains a lot of development code that is not needed for production. In particular, there is code for testing writing of files to be read in fortran

For now, use the same map files as has Phil for Albedo and TI. I look at image files; determine latitude range of valid data.

Albedo is good over lines 21:1416,

TI is good over lines 100:1339, i.e., 100 bad lines at each end.

Topo is good for all lines

So valid lat range for movie is $\pm 90 - 100./8 = 77.5$

The shifts and reverses necessary to get all files aligned are done, and each is written as an unformatted FORTRAN file.

Source files used:

TES bolometric Albedo

web='asu Bastion: /mars/work/prc/global_images/numeric/tes_prc_albedo_numeric_destriped.vic'

filen='/work2/mars/TES/tes_prc_albedo_numeric_destriped.vic'

ppdi=8 ; source resolution

Others:

web='http://pds-geosciences.wustl.edu/missions/mgs/tesspecial.html'

ppdi=8

Thermal Inertia

web='asu Bastion: /mars/work/prc/global_images/numeric/tes_prc_putzig_inertia_numeric_destriped.vic'

filen='/work2/mars/TES/tes_prc_putzig_inertia_numeric_destriped.vic'

ppdi=8 ; source resolution

Others:

web='http://lasp.colorado.edu/inertia/2007/index.html'

filen=/work2/mars/TImaps/NBmap2007.bin ; process inertia

ppdi=20 ; source resolution

Elevation

web='http://pds-geosciences.wustl.edu/missions/mgs/mola.html'

filen=/work1/mars/MOLA/megt90n000eb.img

ppdi=16 ; source resolution

Normal action sequence: @111: kons=[2,212,45,222,45,23,45]

2.... Set output sizes

212.. Process albedo

45... Raw output vvv

222.. Process TI

45... Raw output vvv

23... Process elevation

45... Raw output vvv

4 Thermal model calculation

4.1 new code

Main program **mkrc.f** replaces **krc.f**

mlats.f replaces **tlats.f**, from which it is derived with few real changes except for a loop on all longitudes for a given latitude.

Modify *krccom.inc* to allow some larger sizes; point to it with a soft-link.

Modify the Makefile to include building **mkrc**.

All other parts of the KRC system are unchanged.

New code in either FORTRAN or IDL is:

C_Tit1 MKRC KRC planet surface thermal model. Map movie version

C_Tit1 MLATS Latitude computations for the KRC thermal model system

C_Tit1 READAIE Read line-interleaved file of albedo, inertia and elevation

```

;_Titl  MAPREBIN  Put Albedo, inertia, elevation maps onto uniform grid
;_Titl  KRCTILE   Process mkrc.f output to tiles for thermal movie

```

4.1.1 Main program; mkrc.f

Most size limits are firm-coded in the common **krccom**. The input file is read in the normal KRC way; except that the read-terminating '0/' line is followed three lines of file names which are read directly by **mkrc**

There is an outer loop on latitude bands, each contains 1° and generates an output file labeled with the first (most northern) latitude with 0.1 °resolution. **readaie** is called for each loop

The inner loop does all latitudes in a band. For each latitude it calls **mlats** once with the three geophysical parameters for all longitudes, then decrements the latitude.

4.1.2 Albedo, inertia and elevation for a zone: readaie.f

The directory and file names are passed in from mKRC;

The first argument defines whether to get values for a single latitude or to read all the map files and store the results for a specified range of latitude. The storage limit is firm-coded as 2880 longitudes and 80 latitudes.

4.2 mkrc Input file

Input file example for MKRC

Set DAU and SDEC for desired Ls. Can use IDL routines:

```
jd=L_S(ls,aud,kode=2)
```

```
j0=L_S(2012,kode=3)
```

```
print,j0-2440000+jd,'=DJUL',aud[1],'=SOLARDEC',aud[0],'=DAU',ls,'=LsubS'
```

Set N4=1 single latitude

Set N2 and N24= 2, multiple of 384 Hours output

Set IDISK2 to number pixels per latitude

Set ALAT(1) to first (most northern)latitude.

All other latitude entries are read but ignored.

Set K4OUT=nLatBand to number of 1-degree bands to do

The example below does one band, which is a good idea until success is certain. To do all of +77 to -77, set ALAT(1) to 77.0 and nLatBand to 154

```

0 0 / KOLD: season to start with; KEEP: continue saving data in same disk file
Master for MKRC All with leading x ignored

```

xALBEDO	EMISS	xINERTIA	COND2	DENS2	PERIOD	SpecHeat	DENSITY
.25	1.00	200.0	3.4	928.0	1.0275	630.	1600.
CABR	AMW	[ABRPHA	PTOTAL	FANON	TATM	TDEEP	SpHeat2
0.11	43.5	-0.00	510.0	.055	200.	180.0	1300.
TAUD	DUSTA	TAURAT	TWILI	ACR2	[ARC3	SLOPE	SLOAZI
0.3	.90	0.5	0.0	0.5	-0.00	0.0	90.
TFROST	CFROST	AFROST	FEMIS	AF1	AF2	FROEXT	[FD32
146.0	589944.	.65	0.95	0.54	0.0009	50.	0.0
RLAY	FLAY	CONVF	DEPTH	DRSET	DDT	GGT	DTMAX
1.2000	.1800	3.0000	0.0	0.0	.0020	0.1	0.1
xDJUL	xDELJUL	SOLARDEC	DAU	xLsubS	SOLCON	GRAV	AtmCp
16082.8	1.02749	21.815	1.60850	120.0	1368.0	3.727	735.9
ConUp0	ConUp1	ConUp2	ConUp3	ConLo0	ConLo1	ConLo2	ConLo3
0.013	0.0	0.0	0.0	0.220	0.0	0.0	0.0
SphUp0	SphUp1	SphUp2	SphUp3	SphLo0	SphLo1	SphLo2	SphLo3
630.0	0.0	0.0	0.0	1300.	0.0	0.0	0.0
N1	N2	N3	N4	N5	N24	IB	IC
15	768	15	1	0	768	0	999

```

      NRSET      NMHA      NRUN      JDISK      IDOWN      FlxP14      FlxP15      KPREF
        3         24         1         81         0         45         65         1
nlatBand      JBARE      xNotif      IDISK2=nPix      end
        01         0         20         2880      0
      LP1      LP2      LP3      LP4      LP5      LP6 LPGL0B      LVFA      LVFT      LkofT
        F        T        F        F        F        F        F        F        F        F
      LPORB      LKEY      LSC      spare      LOCAL      Prt76 LPTAVE      Prt78      Prt79      L_ONE
        F        F        F        F        T        F        F        F        F        F
Latitudes: in 10F7.2 -----7 -----7 -----7 -----7 -----7 -----7 -----7
      71.00 -80.00 -70.00 -60.00 -50.00 -40.00 -30.00 -20.00 -10.00  0.00
xElevations: in 10F7.2 -----7 -----7 -----7 -----7 -----7 -----7 -----7
      1.23   2.01   1.39   1.22   0.38   0.48   1.17   1.67   1.26   0.17
0/
"/work2/mars/TES/remap"      / Image input directory and stem
'Alb8.dat'  'Ti8.dat'  'Topo8.dat'      /   3 image names, in order
"/work2/mars/mKRCout/mov"      / Output directory and stem
0/

```

5 Tiling: krctile.pro

This routine also include code to make and display a single frame of a movie, but without projection onto a sphere. It can also “play” a movie without projection.

Timing, and storage requirements, depend on whether scaling to byte is done early (as mKRC files are read in) or late (as tiles are filled). Both are implimented, and selected by @15: item 8.

Plan: Work toward tiles of 1GB, prefer division in longitude, as only half the planet is visible in one frame. BUT, if sub-view point is at latitude that exceeds width of the polar bland band, still need all longitudes for a frame.

@40 will predict required storage requirements

Storage of a zone’s worth of mKRC output files, in vvv:

```

early: bytarr(nhou,nlon,numb*nlat)
late: intarr(nhin,nlon,numb*nlat)

```

Major operational sections

- 41: Read and scale a zone of mkrc output file
- 42: Write output tiles for this zone
- 43: Read tiles for one zone
- 44: Make one frame of movie in simCyl format
- 45: Play movie of a zone in simCyl format

5.0.1 Some size options

First column of sizes is for going to byte late; second column is for early

```

1  768  MRKC files: N times
2    8   " pix/deg
3    8   " rows/band
4    8  Tiles: pix/deg
5   20   " degrees Lat per tile
6    9   " N blocks in longitude
MKRC file Mbyte   35.39
Zone storage "   1327.10  707.79
Tile size      "    147.46
Minor arrays "     0.02
Total shared "   1474.58  855.27

```

```

1 768 MRKC files: N times
2 8 " pix/deg
3 8 " rows/band
4 8 Tiles: pix/deg
5 30 " degrees Lat per tile
6 6 " N blocks in longitude
MKRC file Mbyte 35.39
Zone storage " 1990.66 1061.68 cannot create a 1.9GB array in IDL
Tile size " 331.78
Minor arrays " 0.02
Total shared " 2322.46 1393.48

```

6 Timing results

6.0.2 mkrc

768 times of day, 4 strips of 0.1 deg latitude

```
Case 0 DTIME: total, user, system= 15.8986 15.7196 0.1790
```

768 times of day, bands of 1 degree, each of 10 strips

```
Case 0 DTIME: total, user, system= 32.4961 32.0631 0.4329 1 band
```

```
Case 0 DTIME: total, user, system= 31.9751 31.5342 0.4409 "
```

```
Case 0 DTIME: total, user, system= 31.6972 31.2353 0.4619 "
```

```
Case 0 DTIME: total, user, system= 627.1946 622.4114 4.7833 20 bands
```

```
Case 0 DTIME: total, user, system= 1923.3176 1905.4904 17.8273 60 bands
```

The forecast time required to compute $\pm 77.5^\circ$ is just under 5000 sec, or about 83 minutes

It would be easy to see up a script or routine to generate input files to do this task in strips as small as 1 degree on up to 155 machines, with wall clock time as small as 33 seconds (assuming the same cpu speed as my computer, a CPU AMD Operton 2.4GHz)

6.0.3 KRCTile

40x20

early

```
AAA BYTE = Array[2880, 320, 160]
```

```
VVV BYTE = Array[2880, 2880, 160]
```

```
DD BYTE = Array[2880, 160]
```

```
Seconds 41 98.217547
```

```
Seconds 42 10.656396
```

```
Seconds 43 46.227139
```

```
Seconds 44 0.11831498
```

not early

```
Seconds 41 8.2144771 9.2321889
```

```
Seconds 42 77.739820 73.370485
```

```
Seconds 43 34.485461
```

```
Seconds 44 0.028468847
```

60X30

not early

```
Seconds 41 40.713062 30.377472
```

```
Seconds 42 109.19824
```

```
Seconds 43 61.104452
```

A Some of the email exchanges with Phil

A.0.4 08/02/2011 03:36 PM

[Text for KRC paper]

In an extreme case, a movie of a thermal day on Mars was made by computing surface temperatures at 0.017 hour intervals for a period of 5 sols at L_s 90. The measured albedo, thermal inertia and elevation with 0.0625 deg resolution was used. 1.45 billion instances of KRC were run in parallel on a Dell cluster of 128 CPU's for 48 hours over a weekend. The video can be viewed at http://mars.asu.edu/~phil/ls90_32_full_5_part_movie.mov
[personal communication from Phil Christensen].

Here is what I actually did. For a single L_s (90), I looped through 1440 (4-ppd) ti and alb maps, moving the sun one column each time. For each of these sun positions, I looped through through the 1440 longitudes with a single local time for each. For each of these I constructed a 720 line input file (1 lat, local time, TI, alb, etc) value in 'one.inp' and feed this to krc. So, by my calculations, this is 1440*1440*720 (=1.49 x 10⁹) instances of krc. Alternatively, I only called the krc program 1440*1440 (=2.1x10⁶) times. I'll leave it to you to decide. Each krc call with 720 lines took ~10 sec to complete.

A.0.5 08/17/2011 09:32 AM

Phil.

This is confirmation of our agreement that thermal inertia-related global map arrays are simple-cylindrical and will start at the North-west corner [Long=0, 90N] and the most-rapidly changing index will increment eastward.

Please reply with the resolution in latitude, longitude and time that you have used thus far for your global movies.

Hugh

A.0.6 08/18/2011 05:29 AM

Hugh, I agree with your convention.

I use 8 ppd for the longitude and latitude resolution of the maps, with a corresponding time resolution of 24/2880 or 0.0008333 hours.

The files I use are at:

```
/work/prc/global_images/numeric/tes_prc_putzig_inertia_numeric_destriped.vic
/work/prc/global_images/numeric/tes_prc_albedo_numeric_destriped.vic"
```

Phil

A.0.7 08/18/2011 05:55 AM

Got them, using a /mars in front. Skipped 348 bytes, then read albedo.

I find Olympus Mons near the left side, so you must be using 180 as the first pixel!

Need to chat about this.

Hugh

A.0.8 08/18/2011 06:11 AM

You are correct. In my script I swap lons to 0-360. Sorry for the confusion.
Phil

A.0.9 08/20/2011 04:39 PM

From Phil:

I don't have a preference on ranking - I suspect I'll have to untangle the files to make the flat maps - one for each time step in which the local time varies with longitude across the map.

As for temp scaling, I might prefer integer - in an image having a 0.7 K digitization might be visible.

As for tiling, a 100-200 mbyte file, which I will untangle to flat maps - should be fine. I confess I don't fully understand your output scheme yet. Can you do 10 deg latitude bands with all longitudes and local times in a single file? I clearly need a better idea of how the output will be structured.

B IDL-FORTRAN bug

Found that first word written in IDL must be ==0.

```
C Discovered FORTRAN limitations that require reading the entire file in
C a single read, and first byte or word must be 0. This constrains efficiency
C Files expected to be  R*4 (nsamp,nline)  3=[alb,iner.elev]
C When called with  IK lt 1, opens files, [reades header and skips -i lines],
C Reads entire file, Transfers  nk lines starting after -IK into saved array
C When  IK 1 or more, extracts that relative line from the saved array,
C unscales it into  AAA,RRR,VVV
```

2011sep17: Send test code and output to Mike Wolff, who had volunteered to look at it.