

# USGS Landsat Data Continuity Mission (LDCM) Landsat 8 Cloud Cover Algorithm (CCA) and C Language Function of Mask (CFMask) Cloud Cover Assessment Algorithm Description Document (ADD)

## CCA – CFMask

### Background/Introduction

CFMask is the C language version of the Function of Mask algorithm; a cloud cover assessment (CCA) algorithm originally researched at Boston University and developed further at EROS. CFMask is a multi-pass algorithm that uses decision trees to prospectively label pixels in the scene, then validates or discards those labels according to scene-wide statistics.

CFMask is attractive as a solution to Landsat cloud masking because of its high accuracy and inclusion of a cloud shadow detection algorithm. Refinements in CFMask include the ability to operate without thermal data, the ability to use cirrus band data, and variant algorithms that allow it to run on data from several Landsat-like instruments.

The output of CFMask is an intermediate CCA mask which can be used to create the final scene QA mask.

### Inputs

Descriptions	Data (Units)	Level	Source	Type
Reflectance band scene data, as TOA reflectance	Reflectance (None)	Scene, OLI bands 2-7, 9 or TM/ETM+ bands 1-5 and 7.		float
Thermal band scene data, as TOA Brightness Temperature	Brightness Temperature (Celsius)	Scene, OLI TIRS Band 1 or TM/ETM+ band 6		float
Scene Elevation image	meters	Scene	DEM	long

### Outputs

Each component of CFMask – fill, water, cloud shadow, snow/ice, cloud and/or cloud confidence could potentially be assigned as values (0, 1, 2, etc.) or as bits (00, 01, 10, 11) depending on the needs of downstream application(s).

## Prototype Code

The current CFMask code exists in the GitHub repository at <https://github.com/USGS-EROS/empa-cloud-masking/tree/master/cfmask>.

## Procedure

CFMask involves several passes. In each pass, the algorithm evaluates each non-fill pixel in the scene.

1. Pre-defined variables
  - a.  $t\_buffer = 4.0$
  - b.  $cloud\_prob\_threshold = 22.5$
2. Pass 1 – Basic tests. For each pixel in the scene...
  - a. If any satellite other than Landsat 8, if this pixel is at the TOA saturation value for any band, set that band's pixel value to the TOA maximum value.
  - b. Calculate NDVI and NDSI.
    - i.  $NDVI = (r - nir)/(r + nir)$  (If  $r+nir == 0$ , set  $NDVI = 0.01$ )
    - ii.  $NDSI = (g - swir1)/(g + swir1)$  (If  $g+swir1 == 0$ , set  $NDSI = 0.01$ )
  - c. If any visible band (b, g, r) is saturated, mark pixel as saturated.
  - d. Perform Basic Cloud Test.
    - i. If ( $NDSI < 0.8$   
and  $NDVI < 0.8$   
and  $swir2 > 0.03$ ), then positive result.
    - ii. If result is positive and not using thermal,  
then mark pixel as cloud.
    - iii. If result is positive and using thermal,  
and  $thermal < 27C$ ,  
then mark pixel as cloud.
  - e. If pixel is marked as cloudy...
    - i. Perform whiteness test.
      1. Calculate Visible Mean
        - a.  $visi\_mean = (b + g + r)/3.0$
      2. Calculate Whiteness
        - a. If ( $visi\_mean = 0.0$ ) then whiteness = 100.0
        - b. If any satellite other than Landsat 8, and any visible band (R, G, B) is saturated, then whiteness = 0.0.  
Also set a saturation flag ( $satu\_bv$ ) that will be used in the HOT tests.



- i.  $\text{water\_ptm} = \text{Clear Water Count} / \text{Total Non-Fill Image Pixels}$ .
  - c. Calculate clear land percentage.
    - i.  $\text{land\_ptm} = \text{Clear Land Count} / \text{Total Non-Fill Image Pixels}$ .
  - d. If  $\text{clear\_ptm} \leq 0.1$ , then assume the entire scene is cloudy or cloud shadowed.
    - i. Mark all non-cloudy pixels in scene as cloud shadow.
    - ii. Mark all cloudy pixels in the scene as high-confidence cloud.
    - iii. If using thermal, then disable the thermal thresholds:
      - 1. Set  $t\_templ = -1.0$
      - 2. Set  $t\_temph = -1.0$
  - e. If  $\text{land\_ptm} \geq 0.1$  then expect clear land.
    - i. Set  $\text{land\_bit} = \text{clear\_land\_bit}$ .
    - ii. else set  $\text{land\_bit} = \text{clear\_bit}$ .
  - f. If  $\text{water\_ptm} \geq 0.1$  then expect clear water.
    - i. Set  $\text{water\_bit} = \text{clear\_water\_bit}$ .
    - ii. else set  $\text{water\_bit} = \text{clear\_bit}$ .
- 4. Pass 2 – Calculate Temperature Statistics. For each pixel in the scene...
  - a. If any satellite other than Landsat 8, if this pixel is at the TOA saturation value for any band, set that band's pixel value to the TOA maximum value.
  - b. If pixel is land and using thermal, calculate land brightness temperature.
  - c. If pixel is water and using thermal, calculate water brightness temperature.
  - d. While calculating brightness temperatures, remember the minimum and maximum land brightness temperatures and the minimum and maximum water brightness temperatures in the scene. Also keep tallies of the number of land pixels and water pixels.
  - e. If there are no clear land pixels, set minimum and maximum land temperature to zero.
  - f. If there are no clear water pixels, set minimum and maximum water temperature to zero.
- 5. If using thermal, calculate temperature percentiles.
  - a.  $t\_templ = 17.5\% \text{ percentile land temperature} - t\_buffer$
  - b.  $t\_temph = 82.5\% \text{ percentile land temperature} + t\_buffer$

c.  $t\_wtemp = 82.5\%$  percentile water temperature.

6. Pass 3 – Calculate Cloud Probability. For each pixel in the scene...

a. If any satellite other than Landsat 8, if this pixel is at the TOA saturation value for any band, set that band's pixel value to the TOA maximum value.

b. If pixel is water...

i. Calculate Brightness Probability.

1.  $brightness\_prob = swir1/0.11$ , clipped to between 0.0-1.0.

2. If using thermal,  $wtemp\_prob = (t\_wtemp - thermal\ BT)/4.0$ .

a. If  $wtemp\_prob < 0.0$ , set = 0.0.

b.  $brightness\_prob = brightness\_prob * wtemp\_prob$

3. If using the cirrus band, then  $brightness\_prob = brightness\_prob + cirrus\ reflectance/0.04$ .

ii.  $wfinal\_prob = 100.0 * brightness\_prob$

c. If pixel is land...

i. Calculate modified NDVI and modified NDSI.

1.  $NDVI = (r - nir)/(r + nir)$

a. If  $r+nir == 0$ , set  $NDVI = 0.01$ .

2.  $NDSI = (g - swir1)/(g + swir1)$

a. If  $g+swir1 == 0$ , set  $NDSI = 0.01$ .

3. Clip both NDVI and NDSI to positive values.

a. If less than 0.0, set equal to 0.0.

ii. Calculate whiteness.

1. Calculate Visible Mean

a.  $visi\_mean = (b + g + r)/3.0$

2. Calculate Whiteness

a. If  $(visi\_mean = 0.0)$  then set  $whiteness = 0.0$

b. Else,  $whiteness = (abs(b + g + r - visi\_mean))/visi\_mean$

3. If any satellite other than Landsat 8, and any visible band (R, G, B) is saturated, then  $whiteness = 0.0$ .

iii. Calculate probabilities.

1.  $\text{vari\_prob} = 1.0 - \text{maximum of } (\text{max}(\text{abs}(\text{NDVI}), \text{abs}(\text{NDSI}), \text{whiteness}))$
  2. If using thermal,
    - a.  $\text{temp\_prob} = (\text{t\_temph} - \text{thermal BT}) / (\text{t\_temph} - \text{t\_templ})$ . (If  $\text{temp\_prob} < 0.0$ , set = 0.0.)
    - b.  $\text{vari\_prob} = \text{vari\_prob} * \text{temp\_prob}$
  3. If using cirrus, then  $\text{vari\_prob} = \text{vari\_prob} + \text{cirrus reflectance} / 0.04$ .
  4.  $\text{final\_prob} = 100.0 * \text{vari\_prob}$
- d. Calculate dynamic land cloud threshold.
- i. Threshold  $\text{clr\_mask} = 82.5\%$  percentile value of  $\text{final\_prob}$  pixels that are also marked as  $\text{land\_bit}$ .
  - ii. Add  $\text{cloud\_prob\_threshold}$  to  $\text{clr\_mask}$ .
- e. Calculate dynamic water cloud threshold.
- i. Threshold  $\text{wclr\_mask} = 82.5\%$  percentile value of  $\text{wfinal\_prob}$  pixels that are also marked as  $\text{water\_bit}$ .
  - ii. Add  $\text{cloud\_prob\_threshold}$  to  $\text{wclr\_mask}$ .
7. Pass 4 – Assign confidence levels. For each pixel in the scene...
- a. If using thermal and  $\text{thermal BT} < (\text{t\_templ} + \text{t\_buffer} - 35.0)$ , then mark pixel as high confidence cloud and skip the remaining tests.
  - b. If pixel is water, is marked as cloud, and  $\text{wfinal\_prob} > \text{wclr\_mask}$ , then mark pixel as high confidence cloud.
  - c. If pixel is land, is marked as cloud, and  $\text{final\_prob} > \text{clr\_mask}$ , then mark pixel as high confidence cloud.
  - d. If pixel is water, is marked as cloud, and  $\text{wfinal\_prob} > \text{wclr\_mask} - 10.0$ , then mark pixel as medium confidence cloud.
  - e. If pixel is land, is marked as cloud, and  $\text{final\_prob} > \text{clr\_mask} - 10.0$ , then mark pixel as medium confidence cloud.
  - f. In all other cases, mark pixel as low confidence cloud.
8. Pass 5 – Potential Cloud Shadow mask.
- a. Calculate Flood filling Statistics.
    - i. If any satellite other than Landsat 8, then for each pixel in the scene, if this pixel is at the TOA saturation value for the NIR or

- SWIR1 band, set that band's pixel value to the TOA maximum value.
  - ii. Calculate min and max values for both the nir and swir1 bands.
  - iii. Calculate 17.5% percentile of both nir and swir1 bands.
  - iv. Use percentile values to create a flood-filled image for both nir and swir1.
- b. For each pixel in the scene...
  - i. If any satellite other than Landsat 8, then for each pixel in the scene, if this pixel is at the TOA saturation value for the NIR or SWIR1 band, set that band's pixel value to the TOA maximum value.
  - ii. Shadow probability = minimum of (new nir band or new swir1 band).
  - iii. If shadow probability > 0.02 (in reflectance units) and the pixel is not marked as water, then mark the pixel as potential shadow. Otherwise, mark it as not shadow.

## 9. Cloud Shadow Detection

- a. If scene is less than 10% clear, do not run shadow processing; set all non-cloud pixels to shadow.
- b. Calculate projection angle of clouds to ground.
- c. Map cloud pixels to cloud objects, each containing N pixels.
  - i. Cloud objects with  $N < 9$  pixels are discarded and not used for shadow calculation.
- d. For each cloud object...
  - i. Set thresholds for cloud shadow matching.
    - 1. If cloud is more than 10% of scene area,  $t_{\text{similar}} = 0.1$  and  $t_{\text{buffer}} = 0.98$ . This allows for more lenient matching of large clouds that may have shadows outside the scene borders.
    - 2. Otherwise,  $t_{\text{similar}} = 0.3$  and  $t_{\text{buffer}} = 0.98$ .
  - ii. If using thermal...
    - 1. Calculate the min and max temperature in all pixels of the cloud as  $\text{temp\_obj\_min}$  and  $\text{temp\_obj\_max}$ .
    - 2. Estimate cloud radius:  $\text{rad} = \sqrt{\frac{N}{2 * \pi}}$

3. If cloud radius is less than the minimum valid cloud size ( $rad < num\_pix$ ), then set cloud temperature  $t\_obj = temp\_obj\_min$ .
4. If cloud radius is greater than the minimum valid cloud size, calculate  $pct\_obj = \frac{(rad - 3)^2}{rad^2}$ 
  - a. Calculate  $t\_obj$  as the  $pct\_obj\%$  percentile of the temperature (between  $temp\_obj\_min$  and  $temp\_obj\_max$ ). Example: If  $pct\_obj = 0.5$ , then  $t\_obj$  is at the 50% percentile between the min and max temperatures.
5. Calculate height range.
  - a.  $min\_height = (10.0 * (t\_templ - t\_obj) / 9.8)$ 
    - i. Clip  $min\_height$  to a minimum of 200.
  - b.  $max\_height = 10.0 * (t\_temph - t\_obj)$ 
    - i. Clip  $max\_height$  to a maximum of 12,000.
- iii. If not using thermal, then  $min\_height = 200$  and  $max\_height = 12,000$ .
- iv. Calculate step size.
  1.  $i\_step = (2.0 * pixel\_size * \tan(\text{sun\_elevation}))$
  2. If  $i\_step$  is less than  $2 * pixel\_size$ , clip it to  $2 * pixel\_size$ .
- v. Cloud Height Iteration
  1. For each cloud base height ( $base\_h$ ) from  $min\_height$  to  $max\_height$ , in steps of  $i\_step$ ...
    - a. If using thermal, calculate cloud heights for each pixel in this cloud object:
      - i.  $cloud\_height[x] = (10.0 * (t\_obj - temp\_pixel[x]) / 6.5) + base\_h$
    - b. If not using thermal,  $cloud\_height[x] = base\_h$ .
    - c. Find the true position of this cloud object using this trial height and the scene DEM.
    - d. For each pixel in this cloud, project the pixel onto the ground
      - i.  $i\_xy = \frac{cloud\_height[h]}{pixel\_size * \tan(\text{sun\_elevation})}$
      - ii.  $x' = x + i\_xy * \cos(\text{sun azimuth})$

iii.  $y' = y + i_{xy} + \sin(\text{sun azimuth})$

e. Count up all cloud pixels that are matched with projected ground pixels that are fill, cloud, or preliminary shadow as *match\_all*. Also count the number of pixels projected outside the scene borders (*out\_all*) and the total number of pixels in the cloud (*total\_all*). Add *out\_all* to both *match\_all* and *total\_all*, so as to correct the match weighting for shadows that may be outside the scene.

f. Search for the first maximum with more than *t\_similar%* of the cloud pixels matched with a projected potential shadow pixel on the ground. But allow for a small percentage of variation before settling on a maximum. So:

i. Calculate  $\text{thresh\_match} = \frac{\text{match\_all}}{\text{total\_all}}$

ii. If *thresh\_match* is higher than the record maxima, set the record to equal the *thresh\_match* and remember the cloud heights at this maxima.

iii. If *thresh\_match* is lower than the record maxima but greater than *t\_buffer\*record*, continue with the next trial height. This is to allow small variations around the maxima, in hopes that a greater maxima will be found later.

iv. If *thresh\_match* is lower than the record maxima, lower than *t\_buffer\*record*, and the record maxima is greater than *t\_similar*, then the record maxima is considered best and the recorded cloud heights are correct.

v. If the record maxima is over 0.95, then assume it is the best possible match and the recorded cloud heights are correct.

vi. Once the correct heights are found:

1. Re-calculate the true cloud position using this height and the scene DEM.

2. Re-project the cloud pixels to the ground by recalculating *i\_xy*, *x'*, and *y'*.

3. For every pixel in this cloud object, if its projection onto the ground is marked as potential cloud shadow, mark that pixel as verified cloud shadow.
  4. Escape the height loop and continue with the next cloud object.
- vii. If `thresh_match` is lower than the record maxima and the record maxima is not greater than `t_similar`, continue with the next trial height. This may result in no verified shadow pixels for this cloud object, if the `t_similar` threshold is never reached.

## Verification Methods

The CFMask prototype code is being used to generate cloud masks for a standardized set of data. Masks created by the operational algorithm will be verified by comparison with the prototype masks of the same data set, and by manual inspection with the imagery to verify its accuracy in cloud detection.

## Maturity

Level 1.

Possible changes that may occur in the CFMask algorithm are:

- **Parameter Changes** – The internal parameters and thresholds in the CFMask may undergo further tweaking by the algorithm designers.
- **Optimization** – The CFMask code is not optimized, and should be looked at by a developer with an eye toward improving its performance. It may be possible to combine some passes (pass 1 and 2, or pass 3 and 4) or improve efficiency in other ways.
- **Cirrus options** – This ADD describes a version of the CFMask algorithm that optionally uses the cirrus band in minor calculations. A variant of CFMask may be created that exploits the cirrus band more thoroughly. If this variant is intended for production a separate ADD will be created to describe it.
- **Output format Changes** – The format of the QA band output by the CFMask algorithm may change.
- **Shadow algorithm separability** – The shadow-detection algorithm is a separate piece of code and could be run on any CCA algorithm that outputs a potential cloud shadow mask. This allows the possibility of running a potential future

algorithm in place of the CFMask early passes. No algorithm candidate for doing that currently exists, but the potential is there.

## Notes

### Version 1.07:

Comments to Version 1.05 and answers to questions were noted in Review History Removed header.

Made the overall introduction text less application specific – CFMask can be formatted for numerous output types, not necessarily for a specific QA band.

Ensured units are in Celsius for all code; specified Celsius as expected units for thermal bands.

Added cirrus band to table of inputs.

Removed product table, added short description instead.

Added first step of Procedure section to initiate `t_buffer` and `cloud_prob_threshold` variables with pre-defined values.

Specified that `clear_ptm`, `land_ptm` and `water_ptm` are calculated using all non-fill image pixels.

Explained that “`clr_mask`” and “`wclr_mask`” are calculated using only pixels that are flagged as “`land_bit`” or “`water_bit`”, respectively.

Changed confidence test `a`, `t_tmpl` and `t_tmph` to use `t_buffer` instead of hard-coded value (as `t_buffer` can change.)

Corrected some thresholding typos.

Corrected small typos.

Removed review history.

### Version 1.06:

Internal review comments to Version 1.05 with responses and answers to questions were noted in a section titled “Review History”.

### Version 1.05:

Changed 'high' to 'medium' in step 6.e of Procedure section.

Tim Beckmann suggested bringing the cloud height iteration block (step 8.d.v) out into a new section to improve the formatting, but that doesn't seem possible because it is inside a cloud object loop (step 8.d). No change was made.

### Version 1.04:

Updated description based on slight tweaks to algorithm code (`t_obj = minimum` if `cloud_radius < num_pix`). This corrects some of the questionable behavior described in the version 1.02 notes below.

### Version 1.03:

Added non-thermal and thermal decision points, so that the algorithm can run on data with or without a thermal band. Also added handling for instruments on Landsats 4-7.

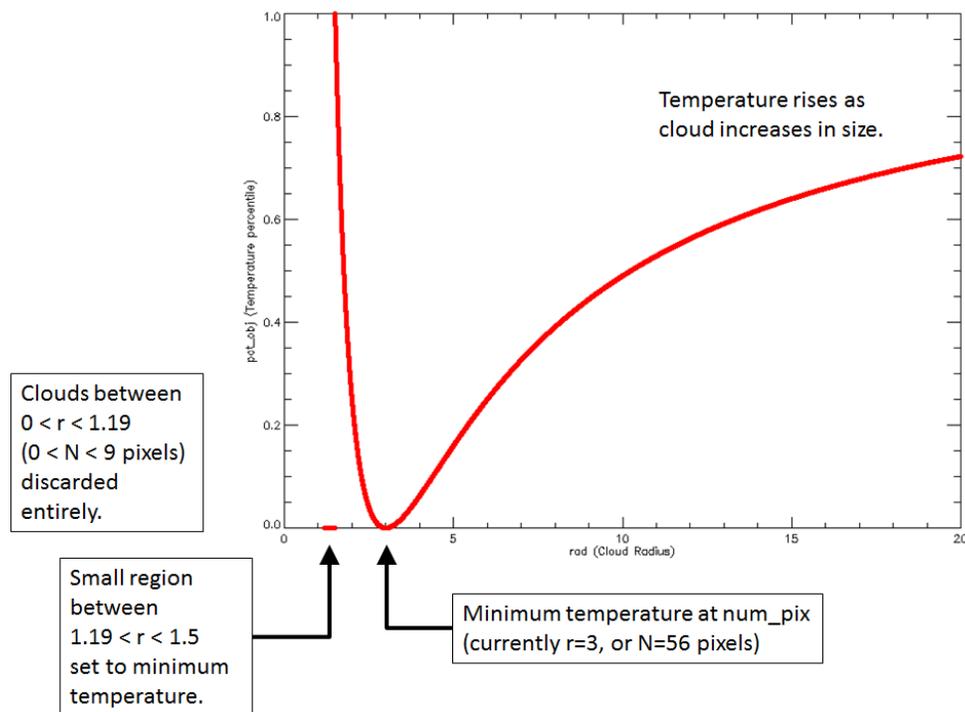
Made changes to resolve questions from version 1.02 (num\_pix should be 3.0, low radius clouds set to minimum temperature). Some questions may still remain and are being discussed. (Should clouds with  $r < 3$  all be clipped to min temp?)

Version 1.02:

Small changes made based on group discussion.

There is one questionable part remaining in the algorithm, the handling of pct\_obj, the temperature percentile of the cloud object. The current behavior is discontinuous, physically nonintuitive, and a comment in the code suggests that a parameter is wrong.

Here is the value of pct\_obj as a function of rad, the radius of the cloud object:



This is questionable because:

- It is discontinuous. For small clouds ( $r < 3$ ) the temperature rises until it reaches maximum, then it is forced to minimum at  $r=1.5$ . The correct behavior would seem to be to set small clouds to maximum temperature. (Zhe Zhu has explained that we cannot trust the instrument to record the temperature correctly for small clouds, so it is safer to assume they are at minimum cloud temperature. That works as an explanation, but it still leaves a discontinuous function.)
- It only appears effective with small clouds. Any cloud with  $r > 3$  pixels will increase in temperature as size increases. This seems physically counter-intuitive, as large clouds should have peaks at higher altitude and lower temperatures. Perhaps this is intended to follow some atmospheric temperature profile.

- c) While the shape of this curve may be intended, a comment in the code suggests that the minimum is at too small a value:

```
float num_pix = 3.0; /* number of inward pixes (240m) for cloud base
temperature */ (object_cloud_shadow_match.c , github code line 361)
```

If num\_pix is intended to be 3 x 240m pixels, then on 30m data it should be a value of 24.0, which gives a minimum temperature at  $r=24$  or  $N=3619$ . This is a much larger cloud, and clouds smaller than that would increase in temperature with decreasing size as expected.

Version 1.01:

Introduction added. Several changes made based on commentary from Steve Foga.

Version 1.0:

Initial release.