```
pro surface_distribution, input, output, npack, seed

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;
;;   Distribute pacets about a sphere with radius r=SpatialDist.exobase
;;
;; Version History
;;   3.2: 12/16/2010
;;     * rewrote the way random points on the surface are chosen
;;   3.1: 11/23/2010
;;     * minor revision in (SpatialDist.use_map EQ 1) section
;;   3.0: 7/19/2010
;;     * rewriting with new strucutre architecture
;;   2.1: Added better support for surface distributions
;;   2.0: File created.
;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;; options: use_map, latitude, longitude
SpatialDist = input.SpatialDist

if (SpatialDist.use_map) then begin
  if ~(file_test(SpatialDist.mapfile)) then stop
  restore, SpatialDist.mapfile
  RandomDeviates_2d, *sourcemap.map, *sourcemap.longitude, sin(*sourcemap.latitude), $
    npack, lon, lat
  lat = asin(lat)
  destroy_structure, sourcemap
endif else begin
  ;; Choose the latitude -- f(lat) = cos(lat)
  if ((SpatialDist.latitude)[0] EQ (SpatialDist.latitude)[1]) $
    then lat = replicate((SpatialDist.latitude)[0], npack) $
    else begin
      sinlat = random_nr(seed=seed, npack)*2.-1
      lat = asin(sinlat)
    endelse

  ;; Choose the longitude -- f(lon) = 1 / (lonmax-lonmin)
  if ((SpatialDist.longitude)[0] GT (SpatialDist.longitude)[1]) $
    then m = [(SpatialDist.longitude)[0], (SpatialDist.longitude)[1]+2*!pi] $
    else m = SpatialDist.longitude
  lon = (m[0] + (m[1]-m[0]) * random_nr(seed=seed, npack)) mod (2*!pi)
endelse

if strcmp(input.geometry.planet, input.geometry.StartPoint, /fold) then begin
  ;; Starting at a planet.
  ;; 0 deg longitude = subsolar pt = (0, -1, 0)
  ;; 90 deg longitude = dusk pt = (1, 0, 0)
  ;; 270 deg longitude = dawn pt = (-1, 0, 0)
  *output.x0 = double(SpatialDist.exobase * sin(lon)*cos(lat))
  *output.y0 = -double(SpatialDist.exobase * cos(lon)*cos(lat))
```

```
52      *output.z0 = double(SpatialDist.exobase * sin(lat))
53    endif else begin
54      ;; Starting at a satellite
55      ;; Treats the satellite as if it were at phi = 0.
56      ;; 0 deg longitude = subsolar pt = (0, -1, 0)
57      ;; 90 deg longitude = leading pt = (-1, 0, 0)
58      ;; 270 deg longitude = trailing pt = (1, 0, 0)
59      ;; lon=0 -> sub-planet point; lon=90 -> leading point
60      *output.x0 = -double(SpatialDist.exobase * sin(lon)*cos(lat))
61      *output.y0 = -double(SpatialDist.exobase * cos(lon)*cos(lat))
62      *output.z0 = double(SpatialDist.exobase * sin(lat))
63    endelse
64
65    q = where(finite(*output.x0) EQ 0, nq) & if (nq NE 0) then stop
66    q = where(finite(*output.y0) EQ 0, nq) & if (nq NE 0) then stop
67    q = where(finite(*output.z0) EQ 0, nq) & if (nq NE 0) then stop
68
69    end
70
```