leigqNEWTON_sphere_validate

Validate and refine a detected eigen-sphere candidate (recommended entry point).

IMPORTANT: In the public toolbox, this function is a thin wrapper that forwards all inputs to `leigqNEWTON_sphere_refine`. Therefore, the calling convention and outputs are the same.

Both VALIDATE and REFINE expect a *case struct* that contains a sphere model in |cases.sph|. If no sphere model is supplied, the verdict will be "NO_SPHERE_MODEL".

Typical pipeline (recommended):

```
[lamAll,lamS,lam0,cls,sph,info] = leigqNEWTON_sphere_detect(A, ...);
```

```
C = struct('A',A,'lamAll',lamAll,'lamSamples',lamS,'cls',cls,'sph',sph,'info',info);
```

```
[A2,lamAll2,resAll,lamS2,resS,sph2,conf] = leigqNEWTON_sphere_validate(C, ...);
```

This page is intentionally **fast** and runs only `leigqNEWTON_sphere_validate` on a small, pre-defined test case. We do **not** run detection/sampling here. The "previous step" (detect/sample) is here emulated by providing: * a candidate sphere model `sph0`, * a small list of approximate candidates `lamAll` and DISTINCT samples `lamS`. The validator then refines candidates, computes residual certificates, and runs a modest grid test.

For more reliable (but slower) settings, see the "RELIABLE profile" section near the end of this page.

The examples require MATLAB built-in quaternion class.

Setup: requirements and path

```
hasQuat = true;
try
    quaternion(0,0,0,0);
catch
    hasQuat = false;
end
if ~hasQuat
    disp('This toolbox requires the built-in MATLAB quaternion class.');
    return
end
if exist('leigqNEWTON_sphere_validate','file') ~= 2
    error('Sphere functions not found on the MATLAB path. Add the toolbox root
folder.');
end
```

Example: documentation-sized, fast

We reuse the same small 2-by-2 example as in doc_leigqNEWTON_sphere_refine. Here we call the user-facing entry point VALIDATE, which forwards to REFINE.

```
qi = quaternion(0,1,0,0);
A = [ quaternion(2,0,0,0),  qi;
     -qi,                   quaternion(2,0,0,0) ];
```

```
s2 = 1/sqrt(2);
lamS = [ quaternion(2, 1, 0, 0);
         quaternion(2,-1, 0, 0);
         quaternion(2, 0, 1, 0);
         quaternion(2, 0,-1, 0);
         quaternion(2, 0, 0, 1);
         quaternion(2, 0, 0,-1);
         quaternion(2, s2, s2, 0);
         quaternion(2,-s2,-s2, 0) ];
epsPert = 1e-3;
lamAll = lamS + quaternion(epsPert, -epsPert, 0, epsPert);

B  = [0 0 0; 1 0 0; 0 1 0; 0 0 1];
p0 = [2;0;0;0];
c3 = [0;0;0];
r  = 1;

sph = struct();
sph.p0       = p0;
sph.basis4x3  = B;
sph.center3   = c3;
sph.radius    = r;
sph.center4   = p0 + B*c3;
sph.center    = quaternion(sph.center4(1), sph.center4(2), sph.center4(3),
sph.center4(4));
sph.inliers   = (1:numel(lamS)).';

fprintf('Note:');
```

```
Note:
```

```
fprintf('\n  Validation is a lengthy process and may take several minutes.');
```

```
  Validation is a lengthy process and may take several minutes.
```

```
fprintf('\n  Therefore, a shortened version with inconclusive results');
```

```
  Therefore, a shortened version with inconclusive results
```

```
fprintf('\n  is called for documentation purposes.\n');
```

```
  is called for documentation purposes.
```

```
C =
struct('A',A,'lamAll',lamAll,'lamSamples',lamS,'sph',sph,'idx',NaN,'seedUsed',NaN,'o
rigin',"doc_sphere_validate");

RefineArgs = {'DoPolish', false, 'Verbose', 0, 'MaxIter', 120, 'TolX', 1e-12,
'TolFun', 1e-12};
[A2, lamAll2, resAll, lamS2, resS, sph2, conf, out] = leigqNEWTON_sphere_validate(
...
```

```
    C, 'Verbose', 1, 'RefineArgs', RefineArgs, 'GridTheta', 4, 'GridPhi', 8,
'MinGridForDecision', 32); %#ok<ASGLU>
```

```
=== leigqNEWTON_sphere_refine: idx=NaN, seed=NaN, origin=doc_sphere_validate ===
Ktot=8, Kdistinct=8, spheres=1
Initial residuals: lamAll  med=3.83e-01  max=1.41e+00 | lamSamples med=3.83e-01  max=1.41e+00
Refined residuals: lamAll  med=1.23e-09  max=6.62e-09 | lamSamples med=7.07e-10  max=3.63e-09
Inliers (n=8): res med=7.07e-10 max=3.63e-09
Inliers on-sphere (refit): devRel med=0.00e+00 max=1.37e-16
Eigenvalue certification (TargetRes=1.0e-14): lamAll pass=0% | lamSamples pass=50%
Sphere decision: INCONCLUSIVE  reliability=LOW (0.20)
  Why: Sphere points fail residual test without refinement; this does not support a continuum at TargetRes.
Grid (no refine): N=32  passRes=0%  passSphere=100%  passBoth=0%
Grid (refined):  N=32  passRes=0%  passSphere=100%  passBoth=0%
Collapse clustering: K=32 attractor(s) (tol=1.0e-10). Largest clusters: [1 1 1 1 1]
```

Nicely printed summaries

```
fprintf('resAll: median = %.3e, max = %.3e\n', median(resAll), max(resAll));
```

```
resAll: median = 1.227e-09, max = 6.618e-09
```

```
fprintf('resS   : median = %.3e, max = %.3e\n', median(resS),   max(resS));
```

```
resS  : median = 7.067e-10, max = 3.628e-09
```

```

disp('sph2 (refined model):');
```

```
sph2 (refined model):
```

```
disp(sph2)
```

```
                 p0: [4×1 double]
           basis4x3: [4×3 double]
            center3: [3×1 double]
             radius: 1
            center4: [4×1 double]
             center: [1×1 quaternion]
            inliers: [8×1 double]
    samplerOriginal: []
      samplerStable: @(theta,phi)local_sphere_sampler(sph,theta,phi)
```

```

disp('conf.sphere (decision + reliability):');
```

```
conf.sphere (decision + reliability):
```

```
disp(conf.sphere)
```

```
            verdict: 'INCONCLUSIVE'
                why: 'Sphere points fail residual test without refinement; this does not support a continuum at Ta
      reliabilityTag: 'LOW'
    reliabilityScore: 0.2000
```

See also

leigqNEWTON_sphere_refine, leigqNEWTON_sphere_detect, leigqNEWTON_sphere_sample