Sphere Hunting

Tries to detect and validate spherical components in the left spectrum.

Typical workflow:

1. sample many left-eigenvalue candidates (Newton runs)
2. detect/fit candidate spheres from DISTINCT samples
3. refine + validate (grid test) to decide sphere-vs-artifact

This page is written to be **documentation-friendly**: -

- Default profile aims to finish quickly (less than a minute), yet produce a reliable decision.
- A heavier profile is provided for research runs.

0) Setup (requirements + path)

```
rng(0,'twister');
hasQuat = true;
try
    quaternion(0,0,0,0);
catch
    hasQuat = false;
end
if ~hasQuat
    disp("This toolbox requires MATLAB's built-in quaternion class (Aerospace
Toolbox).");
    disp("Examples in this page are skipped.");
    return;
end

% Ensure toolbox root is on the path (add only the root, no genpath).
if exist("leigqNEWTON_sphere_detect","file") ~= 2
    thisFile = mfilename("fullpath");
    if ~isempty(thisFile)
        rootGuess = fileparts(fileparts(fileparts(thisFile))); % .../docs/source ->
toolbox root
        if exist(fullfile(rootGuess,"leigqNEWTON_sphere_detect.m"),"file")
            addpath(rootGuess);
            rehash toolboxcache
        end
    end
end
if exist("leigqNEWTON_sphere_detect","file") ~= 2
    error("Sphere functions not found on the MATLAB path. Add the toolbox root
folder.");
end
```

1) Choose a profile (recommended: "DOC")

Profiles control the **work budget** (RunsMax×Restarts) and the **validation grid** size. If you previously observed very long runtimes, reduce RunsMax and/or Restarts.

```
Profile = "DOC";   % "DOC" (default) | "FAST" (smoke) | "FULL" (research)

switch upper(Profile)
    case "FAST"
        % Very quick smoke test. May produce "INCONCLUSIVE" if the validation grid
is too small.
        Collect   = 8;
        RunsMax   = 12;
        Restarts  = 40;
        GridTheta = 6;
        GridPhi   = 12;
        GridMaxPoints = 60;   % <-- may fall below MinGridForDecision (see below)
        MinGridForDecision = 64;

        TargetRes = 1e-12;    % relaxed for smoke tests
        RefineArgs = {"Mode","auto","TargetResMin",1e-12,"Verbose",0};

    case "DOC"
        % Documentation-friendly: usually seconds, and the grid is large enough
        % to avoid the "INSUFFICIENT GRID SIZE" / "INCONCLUSIVE" outcome.
        Collect   = 16;
        RunsMax   = 30;
        Restarts  = 80;
        GridTheta = 10;
        GridPhi   = 20;
        GridMaxPoints = 200;  % >= MinGridForDecision
        MinGridForDecision = 64;

        TargetRes = 1e-13;    % certificate threshold
        RefineArgs = {"Mode","auto","TargetResMin",1e-13,"Verbose",0};

    case "FULL"
        % Research run (can be slow). Use when you need more distinct samples.
        Collect   = 30;
        RunsMax   = 80;
        Restarts  = 200;
        GridTheta = 12;
        GridPhi   = 24;
        GridMaxPoints = 400;
        MinGridForDecision = 128;

        TargetRes = 1e-14;
        RefineArgs = {"Mode","auto","TargetResMin",1e-14,"Verbose",0};

    otherwise
        error("Unknown Profile: %s", Profile);
end
```

```
approxRestarts = RunsMax*Restarts;
approxGrid = min(GridTheta*GridPhi, GridMaxPoints);
fprintf("Profile=%s | Collect=%d | RunsMax=%d | Restarts=%d | approxRestarts=%d |
grid~%d pts\n", ...
    Profile, Collect, RunsMax, Restarts, approxRestarts, approxGrid);
```

```
Profile=DOC | Collect=16 | RunsMax=30 | Restarts=80 | approxRestarts=2400 | grid~200 pts
```

2) A small test matrix with a known spherical component (Huang–So style)

This 2-by-2 example is intentionally small so that documentation runs are fast.

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

```
A = 2×2 quaternion array
     2 + 0i + 0j + 0k     0 + 1i + 0j + 0k
     0 - 1i + 0j + 0k     2 + 0i + 0j + 0k
```

3) Detect: collect samples and fit sphere candidates (from DISTINCT samples)

```
tDetect = tic;
fprintf("\nDetection is fast:");
```

```
Detection is fast:
```

```
fprintf("\n=== leigqNEWTON_sphere_detect ===\n");
```

```
=== leigqNEWTON_sphere_detect ===
```

```
[lamAll, lamS, lam0, cls, sph, info] = leigqNEWTON_sphere_detect( ...
    A, ...
    "Collect",  Collect, ...
    "RunsMax",  RunsMax, ...
    "Restarts", Restarts, ...
    "Seed0",    1, ...
    "Report",   "off");

fprintf("Detect done in %.2fs: Ktot=%d  Kdistinct=%d  spheres=%d\n", ...
    toc(tDetect), numel(lamAll), numel(lamS), numel(sph));
```

```
Detect done in 0.06s: Ktot=16  Kdistinct=16  spheres=1
```

4) Validate / refine (recommended entry point)

If you ever see: "Sphere decision: INCONCLUSIVE ... Insufficient grid size ..." increase GridTheta/GridPhi and/or GridMaxPoints (and keep GridMaxPoints>=MinGridForDecision).

```
fprintf("\nValidation/refining is slow:");
```

```
Validation/refining is slow:
```

```

C = struct();
C.A = A;
C.lamAll = lamAll;
C.lamSamples = lamS;
C.cls = cls;
C.sph = sph;
C.info = info;
C.idx = 1;
C.seedUsed = 1;
C.origin = "doc_SphereHunting";

tVal = tic;
fprintf("\n=== leigqNEWTON_sphere_validate ===\n");
```

```
=== leigqNEWTON_sphere_validate ===
```

```
[A2, lamAll2, resAll, lamS2, resS, sph2, conf, out] = leigqNEWTON_sphere_validate(
...
    C, ...
    "Verbose", 1, ...
    "TargetRes", TargetRes, ...
    "RefineArgs", RefineArgs, ...
    "GridTheta", GridTheta, ...
    "GridPhi", GridPhi, ...
    "GridMaxPoints", GridMaxPoints, ...
    "MinGridForDecision", MinGridForDecision, ...
    "RefineGrid", true);
```

```
=== leigqNEWTON_sphere_refine: idx=1, seed=1, origin=doc_SphereHunting ===
Ktot=16, Kdistinct=16, spheres=1
Initial residuals: lamAll  med=1.07e-15  max=1.89e-11 | lamSamples med=1.07e-15  max=1.89e-11
Refined residuals: lamAll  med=2.72e-17  max=9.33e-17 | lamSamples med=2.72e-17  max=9.33e-17
Inliers (n=16): res med=2.72e-17 max=9.33e-17
Inliers on-sphere (refit): devRel med=3.89e-16 max=6.66e-16
Eigenvalue certification (TargetRes=1.0e-13): lamAll pass=100% | lamSamples pass=100%
Sphere decision: SPHERE_CONFIRMED  reliability=HIGH (1.00)
  Why: Many grid points on the fitted sphere already have small residuals without refinement (continuum evidence).
Grid (no refine): N=200  passRes=100%  passSphere=100%  passBoth=100%
Grid (refined):  N=200  passRes=100%  passSphere=100%  passBoth=100%
Collapse clustering: K=154 attractor(s) (tol=1.0e-10). Largest clusters: [20 20 2 2 2]
```

```

fprintf("Validate done in %.2fs.\n", toc(tVal));
```

```
Validate done in 37.73s.
```

5) Quick interpretation hooks

- resAll / resS: certificate-like values (smaller is better) - sph2: sphere model (empty if no sphere validated) - conf.sphere: decision + reliability score

```
% Residual summary
medRes = median(resAll);
maxRes = max(resAll);
fprintf("resAll: median = %.3e, max = %.3e\n", medRes, maxRes);
```

```
resAll: median = 2.720e-17, max = 9.327e-17
```

```

% Sphere model (struct or empty)
fprintf("sph2:\n");
```

```
sph2:
```

```
disp(sph2)
```

```
           center4: [2.0000 7.0768e-16 2.9432e-13 6.9955e-13]
            center: [1×1 quaternion]
            radius: 1.0000
          basis4x3: [4×3 double]
                p0: [2.5387 -4.1492e-18 -0.7636 0.3559]
           center3: [3×1 double]
           inliers: [16×1 double]
          samples4: [4×1 quaternion]
           sampler: @(theta,phi)local_sphere_sampler(model,theta,phi)
   samplerOriginal: @(theta,phi)local_sphere_sampler(model,theta,phi)
     samplerStable: @(theta,phi)local_sphere_sampler(sph,theta,phi)
```

```

% Sphere confidence (sub-struct)
fprintf("conf.sphere:\n");
```

```
conf.sphere:
```

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

```
            verdict: 'SPHERE_CONFIRMED'
                why: 'Many grid points on the fitted sphere already have small residuals without refinement (conti
     reliabilityTag: 'HIGH'
   reliabilityScore: 1
```

```

% Tip: if Max residual is above TargetRes, either:
%   - increase refinement effort (RefineArgs / TargetResMin), or
%   - relax TargetRes for documentation runs.
```