The EGADS Derivative API

Engineering Geometry Aircraft Design System

at ESP Revision 1.29

Marshall Galbraith & Bob Haimes

galbramc@mit.edu haimes@mit.edu

Aerospace Computational Design Lab

Department of Aeronautics & Astronautics

Massachusetts Institute of Technology

Table of Contents

Overview

Provide geometric parameter sensitivities

- Bottom-up geometry construction fully differentiated
  - Sensitivities stored in ego objects
  - Configuration sensitivities computed with discrete evaluation
- EGADS geometry routines differentiated with operator overloaded automatic differentiation

EGADS API – Geometry_dot

Set a Geometry Object's sensitivity

```
icode = EG_setGeometry_dot(ego object, int oclass, int mtype,
                           const int *ints, const double *reals,
                           const double *reals_dot);
icode = IG_setGeometry_dot(I*8 context, I*4 oclass, I*4 mtype,
                           I*4         ints, R*8            reals,
                           R*8            reals_dot)
      geom.setGeometry_dot(oclass, mtype, reals, reals_dot, ints=None)
```

object	the Geometry Object: NODE, CURVE, or SURFACE
oclass	the Object Class associated with the reals
mtype	the Member Type associated with the reals (depends on `oclass`)
ints	the integer information (if none use NULL)
reals	the original real data used to construct the geometry (cannot be retrieved with EG_getGeometry)
reals_dot	the sensitivity of the reals to set in the object
icode	the integer return code

Notes: `ints` is required for either `mtype` = BEZIER or BSPLINE.
Calling `EG_setGeometry_dot` with `ints`, `reals` & `reals_dot` set to NULL clears the object of all sensitivities (`oclass` and `mtype` may be 0).

EGADS API – Geometry_dot

Create a Geometry Object with sensitivities

```
icode = EG_makeGeometry_dot(ego context, int oclass, int mtype,
                            ego rGeom, const int *ints,
          const double *reals, const double *reals_dot, ego *nGeom);
icode = IG_makeGeometry_dot(I*8 context, I*4 oclass, I*4 mtype,
                            I*8 rGeom, I*4         ints,
          R*8            reals, R*8            reals_dot, I*8  nGeom)
nGeom = context.makeGeometry_dot(oclass, mtype, reals, reals_dot,
                                 ints=None, geom=None)
```

context	the Context Object
oclass	the Object Class: PCURVE, CURVE or SURFACE
mtype	the Member Type (depends on `oclass`)
rGeom	the reference Geometry Object (if none use NULL)
ints	the integer information (if none use NULL)
reals	the real data used to construct the geometry
reals_dot	the sensitivity of the real data
nGeom	the returned pointer to the new Geometry Object
icode	the integer return code

Notes: This is equivalent to calling EG_makeGeometry followed by EG_setGeometry_dot.

EGADS API – Geometry_dot

Return a Geometry Object's sensitivities

```
icode = EG_getGeometry_dot(ego object,
                           double **reals, double **reals_dot);
icode = IG_getGeometry_dot(I*8 object,
                           R*8      reals, R*8      reals_dot)
        reals, reals_dot = object.getGeometry_dot()
```

object	the Geometry Object with sensitivities: NODE, CURVE, or SURFACE
reals	the returned pointer to real data used to describe the geometry (*freeable*)
reals_dot	the returned pointer to sensitivity of the real data (*freeable*)
icode	the integer return code

Query an Object's sensitivities

```
icode = EG_hasGeometry_dot(ego object);
icode = IG_hasGeometry_dot(I*8 object)
        bool = object.hasGeometry_dot()
```

object	the Object to query
icode	the integer return code (populated is EGADS_SUCCESS, missing sensitivities is EGADS_NOTFOUND)

Notes: Checks if all entities in the object hierarchy are populated with sensitivity information

EGADS API – Geometry_dot

Copy and optionally Transform the Object's sensitivities

```
icode = EG_copyGeometry_dot(const ego object,
                            double* mat, double* mat_dot, ego copy);
icode = IG_copyGeometry_dot(I*8       object,
                            I*8     mat, I*8     mat_dot, I*8 copy)
        object.copyGeometry_dot(copy, mat=None, mat_dot=None)
```

object	the Object to copy sensitivities from
mat	the 12 values of the translation/rotation matrix, NULL for a strict copy
mat_dot	the sensitivities of mat, may be NULL for mat without sensitivities
copy	the Object populated with the copied sensitivities
icode	the integer return code

Note: The mat transformation should be consistent with what was used for `EG_copyObject` to create the "copy" Object

Zero out an Object's sensitivities

```
icode = EG_zeroGeometry_dot(ego object);
icode = IG_zeroGeometry_dot(I*8 Object)
        object.zeroGeometry_dot()
```

object	the Geometry or Topology Object
icode	the integer return code

EGADS API – Geometry_dot

Evaluate with sensitivities on an Object

```
icode = EG_evaluate_dot(ego object,
                        double *params, double *params_dot,
                        double *result, double *result_dot);
icode = IG_evaluate_dot(I*8 object,
                        R*8     params, R*8     params_dot,
                        R*8     result, R*8     result_dot)
        result, result_dot = object.evaluate_dot(params, params_dot)
```

object	the input Object
params	NODE – ignored (can be NULL); CURVE, EDGE, EEDGE – the t value SURFACE, FACE, EFACE – u then v
params_dot	the params sensitivity, may be NULL for params without sensitivities
result	the returned position, 1^{st} and 2^{nd} derivatives (see EG_evaluate)
result_dot	the returned position, 1^{st} and 2^{nd} derivatives sensitivities
icode	the integer return code

EGADS API – Geometry_dot

Inverse evaluation on the Object with sensitivities

```
icode = EG_invEvaluate_dot(ego object, double *pos, double *pos_dot,
                           double *params, double *params_dot,
                           double *result, double *result_dot);
icode = IG_invEvaluate_dot(I*8 object, R*8     pos, R*8     pos_dot,
                           R*8     params, R*8     params_dot,
                           R*8     result, R*8     result_dot)
params, params_dot, result, result_dot = object.invEvaluate_dot(pos)
```

object	the input Object
pos	is $[u, v]$ for a PCURVE and $[x, y, z]$ for all others
pos_dot	the pos sensitivity (may be NULL)
params	the returned parameter(s) found for the nearest position on the Object: for PCURVE, CURVE, EDGE or EEDGE the one value is t for SURFACE, FACE or EFACE the 2 values are u then v
params_dot	the returned sensitivity of params
result	the closest position found is returned: $[u, v]$ for a PCURVE (len = 2) $[x, y, z]$ for all others (len = 3)
result_dot	the returned sensitivity of result
icode	the integer return code

EGADS API – Geometry_dot

Populate a *skinning* Surface with sensitivities

```
icode = EG_skinning_dot(ego object, int nCurve, ego *curves);
icode = IG_skinning_dot(I*8 object, I*4 nCurve, I*8  curves)
        object.skinning_dot(curves)
```

object	the *skinned* BSpline surface
nCurve	the number of BSpline curves
curves	the vector of egos curves used to create the *skinned* surface (populated with sensitivity information)
icode	the integer return code

EGADS API – Geometry_dot

Populate an *approximated* Object with sensitivities

```
icode = EG_approximate_dot(ego bspline, int mDeg, double tol,
                           const int *sizes, const double *xyzs,
                           const double *xyzs_dot);
icode = IG_approximate_dot(I*8 bspline, I*4 mDeg, R*8 tol,
                           I*4         sizes, R*8           xyzs,
                           R*8            xyzs_dot)
        bspline.approximate_dot(sizes, xyzs, xyzs_dot,
                                mDeg=0, tol=1.e-8)
```

bspline	the B-spline Object crated with `approximate`
mDeg	the boundary condition used by EGADS [0-2], must be consistent with value used to create bspline
tol	the is the tolerance to use for the BSpline approximation procedure, must be consistent with value used to create bspline
sizes	a vector of 2 integers that specifies the size and dimensionality of the data. If the second is zero, then a CURVE is fit and the first integer is the length of the number of $[x, y, z]$ triads. If the second integer is nonzero, then the input data reflects a 2D map.
xyzs	the data to fit (3 times the number of points in length)
xyzs_dot	the sensitivity of xyzs
icode	the integer return code

EGADS API – Topology_dot

Create a Topology Object with sensitivities

```
icode = EG_makeTopology_dot(ego context, ego geom, int oclass,
                int mtype, double *reals, double *reals_dot,
                int nchild, ego *children, int *senses, ego *topo);
icode = IG_makeTopology_dot(I*8 context, I*8 geom, I*4 oclass,
                I*4 mtype, R*8     reals, R*8     reals_dot,
                I*4 nchild, I*8  children,I*4  senses, I*8  topo)
 topo = context.makeTopology_dot(oclass, mtype=0, geom=None,
                reals=None, reals_dot=None, children=None, senses=None)
```

context	the Context Object
geom	the reference Geometry Object (if none use NULL)
oclass	the Object Class: NODE, EDGE, LOOP, FACE, SHELL, BODY or MODEL
mtype	the Member Type (depends on `oclass` – see EGADS API doc)
reals	the real data: may be NULL except for NODE or EDGE
reals_dot	the sensitivity of the real data: cannot be NULL for NODE or EDGE
nchild	number of children (lesser) Topological Objects
children	vector of children objects (`nchild` in length) with sensitivities
senses	a vector of integer senses for the children (required for FACES & LOOPs only)
topo	the returned pointer to the new Topology Object with sensitivities
icode	the integer return code

Note: EG_makeTopology will also preserve sensitivities in Children

EGADS API – Topology_dot

Query a Topology Object

```
icode = EG_getTopology_dot(ego topo, ego *geom, int *oclass,
                           int *mtype, double *reals, double *reals_dot,
                           int *nchild, ego **children, int **senses);
icode = IG_getTopology_dot(I*8 topo, I*8  geom, I*4  oclass,
                           I*4  mtype, R*8     reals, R*8     reals_dot,
                           I*4  nchild, I*8   children, I*4   senses)
oclass,mtype,geom,reals,reals_dot,children,senses = topo.getTopology_dot()
```

topo	the Topology or *Effective Topology* Object to query
geom	the returned reference Geometry Object (can be NULL)
oclass	the returned Object Class: Topology or *Effective Topology*
mtype	the returned Member Type (depends on oclass)
reals	the real data (at most 4 doubles are filled): NODE – contains the $[x, y, z]$ location, EDGE where the t_{min} and t_{max} (the parametric bounds) are returned and FACE where the $[u, v]$ box is filled → the limits first for u then for v (4 in length)
reals_dot	the returned parametric sensitivity of reals
nchild	the returned number of children (lesser) Topological Objects
children	the returned pointer to a vector of children objects (nchild in length) if a LOOP with a reference SURFACE, then 2*nchild in length (PCurves follow) if a MODEL – nchild is the number of Body Objects, mtype the total ego count
senses	a vector of senses for the children (LOOPs) or inner/outer for (FACEs & SHELLs)
icode	the integer return code (EGADS_OUTSIDE for an open EBody)

EGADS API – Topology_dot

Set range sensitivity

```
icode = EG_setRange_dot(ego object, int oclass,
                        double *range, double *range_dot);
icode = IG_setRange_dot(I*8 object, I*4 oclass,
                        R*8     range, R*8     range_dot)
        object.setRange_dot(range, range_dot)
```

object	the input Object (EDGE)
oclass	the Object Class associated with the range
range	EDGE – 2 vales are filled: t_{start} and t_{end}
range_dot	the sensitivity for range
icode	the integer return code

EGADS API – Topology_dot

Returns the range sensitivity and periodicity

```
icode = EG_getRange_dot(ego object, double *range, double *range_dot,
                        int *periodic);
icode = IG_getRange_dot(I*8 object, R*8     range, R*8     range_dot,
                        I*4  periodic)
        range, range_dot, periodic = object.getRange_dot()
```

object	the input Object (EDGE)
range	EDGE – 2 vales are filled: t_{start} and t_{end}
range_dot	the sensitivity for range
periodic	0 for non-periodic, 1 for periodic in t or u, 2 for periodic in v (or-able)
icode	the integer return code

EGADS API – Topology_dot

Populate a simple Solid Body /w Sensitivities – Incomplete

```
icode = EG_makeSolidBody_dot(ego body, int stype, const double *data,
                             const double *data_dot);
icode = IG_makeSolidBody_dot(I*8 body, I*4 stype, R*8          data,
                             R*8           data_dot)
        body.makeSolidBody_dot(stype, data, data_dot)
```

body the Object created with `makeSolidBody`

stype one of: BOX, SPHERE, CONE, CYLINDER, TORUS

data length and fill depends on `stype`:

BOX	6	$[x, y, z]$ then $[dx, dy, dz]$ for the size of box
SPHERE	4	$[x, y, z]$ of center then the radius
CONE	7	apex $[x, y, z]$, base center $[x, y, z]$, then the radius
CYLINDER	7	2 axis points and the radius
TORUS	8	$[x, y, z]$ of center, direction of rotation, then the major radius and minor radius

data_dot the sensitivities of data

icode the integer return code

EGADS API – Topology_dot

Populate a Face Object with sensitivities

```
icode = EG_makeFace_dot(ego face, ego object,
                        const double *rdata, const double *rdata_dot);
icode = IG_makeFace_dot(I*8 face, I*8 object,
                        R*8           rdata, R*8           rdata_dot)
        face.makeFace_dot(object, rdata=None, rdata_dot=None)
```

face	the resultant returned topological Face Object created with EG_makeFace
object	the Loop populated with sensitivities that was used with EG_makeFace
rdata	may be NULL for Loops
rdata_dot	the sensitivity of rdata
icode	the integer return code

EGADS API – Tessellation_dot

Returns the Discrete Mass Properties with sensitivities

```
icode = EG_tessMassProps_dot(const ego tess, double *xyzs_dot,
                             double *props, double *props_dot);
icode = IG_tessMassProps_dot(I*8       tess, R*8      xyzs_dot,
                             R*8      props, R*8      props_dot)
volume, volume_dot, aeraOrLen, aeraOrLen_dot, CG, CG_dot, I, I_dot
      = tess.tessMassProps(xyzs_dot=None)
```

tess	the Body Tessellation Object used to compute the Mass Properties
xyzs_dot	sensitivities of xyzs on the Tessellation Object (evaluate from body if NULL)
props	14 doubles filled reflecting Volume, Area (or Length), Center of Gravity (3) and the inertia matrix at CG (9)
props_dot	14 doubles with sensitivities of props
icode	the integer return code

Computes and returns the physical and inertial properties of a Tessellation Object.

EGADS API – Top-Down Build Functions

Populate a *Revolved* Body with sensitivities – Incomplete

```
icode = EG_rotate_dot(ego body, const ego src,
                      double angle, double angle_dot,
                      double *axis, double *axis_dot);
icode = IG_rorate_dot(I*8 body, I*8       src,
                      R*8    angle, R*8    angle_dot,
                      R*8     axis, R*8     axis_dot)
        object.rotate_dot(src, angle, angle_dot, axis, axis_dot)
```

body	the Body Object created with `rotate`
src	the source Object (populated with sensitivities) used to create body
angle	the angle used to rotate the object through [0-360 Degrees]
angle_dot	the angle sensitivity
axis	pointer to a point (on the axis) and a direction (6 in length)
axis_dot	pointer to a axis sensitivities (6 in length)
icode	the integer return code

EGADS API – Top-Down Build Functions

Populate an *Extruded* Body with sensitivities – Incomplete

```
icode = EG_extrude_dot(ego body, const ego src,
                       double dist, double dist_dot,
                       double *dir, double *dir_dot);
icode = IG_extrude_dot(I*8 body, I*8        src,
                       R*8    dist, R*8     dist_dot,
                       R*8     dir, R*8      dir_dot)
        body.extrude_dot(src, dist, dist_dot, dir, dir_dot)
```

body	the Body Object created with `extrude`
src	the source Object (populated with sensitivities) used to create body
dist	the distance to extrude
dist_dot	the distance sensitivity
dir	pointer to the vector that is the extrude direction (3 in length)
dir_dot	dir sensitivity (3 in length)
icode	the integer return code

EGADS API – Top-Down Build Functions

Populates a *Ruled* Body with Sensitivities

```
icode = EG_ruled_dot(ego body, int nSection, ego *sections);
icode = IG_ruled_dot(I*8 body, I*4 nSection, I*8  sections)
        body.ruled_dot(sections)
```

nSection	the number of Sections in the *rule* Operation interior repeated sections are ignored
sections	the array of sections (populated with sensitivities) used to create the *rule* body
icode	the integer return code

EGADS API – Top-Down Build Functions

Populates a *Blended* Body with Sensitivities

```
icode = EG_blend_dot(ego body, int nSection, ego *sections,
                     double *rc1, double *rc1_dot,
                     double *rc2, double *rc2_dot);
icode = IG_blend_dot(I*4 nSection, I*8  sections,
                     R*8     rc1, R*8     rc1_dot,
                     R*8     rc2, R*8     rc2_dot)
        body.blend_dot(sections, rc1=None, rc1_dot=None,
                                 rc2=None, rc2_dot=None)
```

body	the Body Object created with the *blend* Operation
nSection	the number of Sections in the *blend* Operation interior sections can be repeated once for C^1 or twice for C^0
sections	the array of sections (populated with sensitivities) used to create the *rule* body
rc1	specifies treatment at the first section (or NULL for no treatment)
rc1_dot	sensitivity of rc1
rc2	specifies treatment at the last section (or NULL for no treatment)
rc2_dot	sensitivity of rc2
icode	the integer return code

EGADS Sensitivity API Index