NLPAPI: An API to Nonlinear Programming Problems. **User's Guide**

Michael E. Henderson

IBM Research Division

T. J. Watson Research Center

Yorktown Heights, NY 10598

mhender@watson.ibm.com

June 25, 2003

1 Introduction

This API provides a way to create and access Nonlinear Programming Problems of the form

minimize $O(\mathbf{v}$

(i.e. LANCELOT's form). For example

$$O(\mathbf{v}) = \sum_{i=1}^{ng} \frac{1}{s_i} g_i(\sum_{j=1}^{ne_i} w_{ij} f_{ij}(R\mathbf{e}_{ij}) + <a_i, \mathbf{v}> -b_i)$$

The $g_i : \mathbb{R} \to \mathbb{R}$ are caled groups functions, the $f_{ij} : \mathbb{R}^{e_{ij}} \to \mathbb{R}$ are called nonlinear element2/F2lement

2.1.1 The Objective

Each problem has an objective function. This can be set with the `NLPSet-Objective` or `NLPSetObjectiveByString` routine, or can be built as the sum of a number of groups.

```
NLPProblem P;
char name[]="Obj";
int nv;
int *v;
```

return. ddF

functions. This is a convenience, but if the derivatives are approximated by differencing it can also substantially reduce the number of operations needed to approximate the derivatives of the objective.

The objective can be evaluated using the routines:Thyoreppndlded ppsitdasthedjectostianerjec50 -1re2 7neb-27(y)240pc1(ppnsc)-1ctsreTdjecJ/F21 11.955 Tf

```
NLPSetInequalityConstraintGroupA
NLPSetInequalityConstraintGroupB
NLPAddNonlinearElementToInequalityConstraintGroup
```

```
nv=3; v[0]=3; v[1]=10; v[2]=9;
l =1. ; u=10. ;
rc=NLPAddEqualityConstraint(P, name, l , u, nv, v, F, dF, ddF,
                                          data, freedata);
```

The data variable allows the user to associate a data block with the objective, that is passed to the functions when they are evaluated. The freedata routine is called when the problem is free'd, so that the data block can be released. The array v lists the nv

```
int c;
```

Inequalities are sometimes dealt with by introducing extra variables called slacks. That is,

$$l \le f(\mathbf{v}) \le u$$

is replaced by an equality constraint and simple bounds on the slack –

$$f(\mathbf{v}) - s = 0$$
$$0 \le s \le u - l$$

These operations take a problem with inequalitiy and equality constraints and convert it to a problem with only equalitiy constraints –

```
NLPConvertToEqualityAndBoundsOnly(P);
```

Finally, equalities are sometimes eliminated by introducing a quadratic penalty and Lagrange multipliers. That is,

$$\begin{aligned}
\text{minimize} \quad & O(\mathbf{v}) \\
\text{subject to} \quad & f(\mathbf{v}) = 0
\end{aligned}$$

becomes

$$\text{minimize} \quad O(\mathbf{v}) +$$

or each, what the original constant element and group scale were. Therefore the routine which creates the terms in the objective requires arrays that it can store this information in –

```
int    g[nc];  /* the ids of the added groups */
double mu;
double l[nc];  /* Lagrange multipliers */
double b[nc];  /* Constant elements */
double s[nc];  /* Group scales */

nc=NLPGetNumberOfEqualityConstraints(P);
```

```
NLPSetObjectiveGroupFunction(g,gf);
NLPSetEqualityConeaalneaalGriupin(;
NLPSetObjectiveGroLcqueon(s;
```

```
ef=NLCreateElementFunctionWithInitialHessian(P,"etype",
                                             n,R,F,dF,ddF,
                                             data,freedata,
                                             ddF0);

ef=NLCreateElementFunctionByString(P,"etype",n,R,
                                   "[x,y,z,w]",
                                   "x**2+y**2-z*w");
```

Here, n is the number of element variables, R the range transformation (or NULL), F, dF, and ddF are routines which evalute F and its derivatives (ddF may be NULL). If ddF is NULL, ddF0 gives an initial guess at the Hessian

```
NLPAddNonlinearElementToEqualityConstraintGroup(P, c, g, w, N):
NLPAddNonlinearElementToInequalityConstraintGroup(P, c, g, w, N):
```

2.1.8 Matrices

NLMatrices are similar to the NLVectors. There are dense matrices, and two

The severity is 4, 8 or 12, the Routine is the routine which issued the error, and the line and file give the line of source code where it was issued. The

4 Example

We will develop the code for creating and solving HS65. HS65 is the problem:

minimize
$$(x_1 - x_2)^2 + (x_1 + x_2 - 10)^2/9 + (x_3 - 5)^2$$
subject to
$$-4.5 \quad x_1 \quad 4.5$$
$$-4.5$$

```
NLPSetSimpleBounds(P, 1, -4.5, 4.5);
```

function as the group function, but element functions, unlike groups, which take a scalar argument, take a vector as argument.

First we include the API prototypes:

```
#include <NLPAPI.h>
```

```
rc=NLPSetObjectiveGroupA(P,group,a);
rc=NLPSetObjectiveGroupB(P,group,5.);
NLFreeVector(a);
```

 Next come bounds on the variables:

```
rc=NLPSetSimpleBounds(P,0,-4.5,4.5);
rc=NLPSetSimpleBounds(P,1,-4.5,4.5);
rc=NLPSetSimpleBounds(P,2,-5.,5.);
```