

NAG Fortran Library, Mark 25  
FLL6I25DCL - License Managed  
Linux 64 (Intel 64 / AMD64), Intel Fortran, Double Precision, 32-bit integers

ユーザーノート

内容

1. イントロダクション .....	1
2. 追加情報 .....	1
3. 一般情報 .....	2
3.1. ライブラリのリンク方法 .....	4
3.1.1. C または C++ からのライブラリの呼び出し .....	6
3.2. インターフェースブロック .....	7
3.3. Example プログラム .....	9
3.4. Fortran 型と強調斜体文字の解釈 .....	11
3.5. メンテナンスレベル .....	12
4. ルーチン固有の情報 .....	13
5. ドキュメント .....	17
6. サポート .....	18
7. コンタクト情報 .....	18

## 1. イントロダクション

本ユーザーノートは、NAG Fortran Library, Mark 25 – FLL6I25DCL（ライブラリ）のご利用方法（リンク方法）を説明します。

本ユーザーノートには、NAG Library Manual, Mark 25（ライブラリマニュアル）には含まれない製品毎の情報が含まれています。ライブラリマニュアルに「ユーザーノート参照」などと書かれている場合は、本ユーザーノートをご参照ください。

ライブラリルーチンのご利用にあたり、以下のドキュメントを必ずお読みください。

- (a) Essential Introduction （ライブラリについての基本的なドキュメント）
- (b) Chapter Introduction （チャプター毎のドキュメント）
- (c) Routine Document （ルーチン毎のドキュメント）

## 2. 追加情報

本ライブラリの動作環境やご利用方法についての最新の情報は、以下のウェブページをご確認ください。

<http://www.nag.co.uk/doc/inun/fl25/fl6idcl/supplementary.html>

### 3. 一般情報

本ライブラリは、Intel® Math Kernel Library for Linux (MKL) が提供する BLAS/LAPACK ルーチンを利用するスタティックライブラリおよび共有ライブラリと、NAG が提供する BLAS/LAPACK ルーチンを利用するスタティックライブラリおよび共有ライブラリを提供します。

本ライブラリは、MKL version 11.2.0 を用いてテストされています。MKL version 11.2.0 は本製品の一部として提供されます。MKL の詳細については Intel 社のウェブサイト <https://software.intel.com/intel-mkl> をご参照ください。

パフォーマンス面からは、MKL を利用するバージョンの NAG ライブラリ `libnag_mkl.a`, `libnag_mkl.so` のご利用を推奨します。これらのライブラリは NAG が提供する BLAS/LAPACK ルーチンを含みません。

また、MKL を利用しないバージョンの NAG ライブラリ `libnag_nag.a`, `libnag_nag.so` が提供されます。これらのライブラリは NAG が提供する BLAS/LAPACK ルーチンを含んでいます。

本ライブラリのほとんどのルーチンはスレッドセーフです。

スレッドセーフではないルーチンについては、ライブラリマニュアルの “Thread Safety” ドキュメントの「3 Lists of Thread Unsafe Routines」をご確認ください。

本製品で提供される MKL version 11.2.0 は、環境変数 `OMP_NUM_THREADS` が設定されていない場合、複数のプロセッサまたはマルチコアチップを持つシステムでは、計算速度の向上のためにマルチスレッドで計算を行います。もし、MKL に複数のプロセッサまたはマルチコアを使わせたくない場合は、`OMP_NUM_THREADS` に 1 を設定してください。例えば、

C シェルの場合：

```
setenv OMP_NUM_THREADS 1
```

Bourne シェルの場合：

```
OMP_NUM_THREADS=1  
export OMP_NUM_THREADS
```

本ライブラリは MKL 10.3 よりも古いバージョンとは互換性がありません。

また、MKL には、条件付きビット単位の再現性 (Bit-wise Reproducibility (BWR)) オプションがあります。

ユーザーコードが一定の条件 (<https://software.intel.com/en-us/node/528579> 参照) を満たしていれば、環境変数 MKL\_CBWR を設定することにより BWR が有効になります。詳細は MKL のドキュメントをご参照ください。しかしながら、多くの NAG ルーチンはこれらの条件を満たしていません。従って、MKL を利用するバージョンの NAG ライブラリの全ルーチンに対して、異なる CPU アーキテクチャに渡り MKL\_CBWR による BWR を保証することはできません。BWR に関するより一般的な情報は、Essential Introduction の「3.13 Bit-wise Reproducibility (BWR)」をご参照ください。

### 3.1. ライブラリのリンク方法

本セクションでは [INSTALL\_DIR] に本ライブラリがインストールされていることが前提となります。

デフォルトの [INSTALL\_DIR] は（システムに依りますが） /opt/NAG/fll6i25dcl もしくは /usr/local/NAG/fll6i25dcl となります。

また、インストール時に [INSTALL\_DIR] を指定することもできます。

- /opt/NAG/fll6i25dcl が在れば、これが [INSTALL\_DIR] になります。
- /usr/local/NAG/fll6i25dcl が在れば、これが [INSTALL\_DIR] になります。
- または、インストール時に指定したインストール先が [INSTALL\_DIR] になります。

MKL を利用するバージョンの NAG ライブラリを利用する場合は、以下のようにリンクを行ってください。（ここで driver.f90 がユーザープログラムです。）

スタティックライブラリを利用する場合：

```
ifort -I[INSTALL_DIR]/nag_interface_blocks driver.f90 \  
      [INSTALL_DIR]/lib/libnag_mkl.a \  
      -Wl,--start-group [INSTALL_DIR]/mkl_intel64_11.2.0/lib/libmkl_intel_lp64.a \  
      [INSTALL_DIR]/mkl_intel64_11.2.0/lib/libmkl_intel_thread.a \  
      [INSTALL_DIR]/mkl_intel64_11.2.0/lib/libmkl_core.a -Wl,--end-group \  
      -liomp5 -lpthread -lm
```

共有ライブラリを利用する場合：

```
ifort -I[INSTALL_DIR]/nag_interface_blocks driver.f90 \  
      [INSTALL_DIR]/lib/libnag_mkl.so \  
      -L[INSTALL_DIR]/mkl_intel64_11.2.0/lib \  
      -lmkl_intel_lp64 -lmkl_intel_thread -lmkl_core \  
      -liomp5 -lpthread -lm
```

MKL を利用しないバージョンの NAG ライブラリを利用する場合は、以下のようにリンクを行ってください。（ここで driver.f90 がユーザープログラムです。）

スタティックライブラリを利用する場合：

```
ifort -I[INSTALL_DIR]/nag_interface_blocks driver.f90 \  
      [INSTALL_DIR]/lib/libnag_nag.a
```

共有ライブラリを利用する場合：

```
ifort -I[INSTALL_DIR]/nag_interface_blocks driver.f90 \  
      [INSTALL_DIR]/lib/libnag_nag.so
```

共有ライブラリを利用する場合には、環境変数 LD\_LIBRARY\_PATH を正しく設定し、実行時のリンクが行えるようにしてください。

C シェルの場合：

```
setenv LD_LIBRARY_PATH [INSTALL_DIR]/lib:[INSTALL_DIR]/mkl_intel64_11.2.0/lib
```

または、既存の設定がある場合には次のように拡張します。

```
setenv LD_LIBRARY_PATH \  
      [INSTALL_DIR]/lib:[INSTALL_DIR]/mkl_intel64_11.2.0/lib:${LD_LIBRARY_PATH}
```

Bourne シェルの場合：

```
LD_LIBRARY_PATH=[INSTALL_DIR]/lib:[INSTALL_DIR]/mkl_intel64_11.2.0/lib  
export LD_LIBRARY_PATH
```

または、既存の設定がある場合には次のように拡張します。

```
LD_LIBRARY_PATH=\  
      [INSTALL_DIR]/lib:[INSTALL_DIR]/mkl_intel64_11.2.0/lib:${LD_LIBRARY_PATH}  
export LD_LIBRARY_PATH
```

注意：

ご利用の環境に依っては、その他のパスも LD\_LIBRARY\_PATH に設定する必要があるかもしれません。（例えば、より新しいバージョンのコンパイラでは、コンパイラの実行時ライブラリのパスが必要となるかもしれません。）

注意：

異なるコンパイラを使用している場合は、ディレクトリ [INSTALL\_DIR]/rtl に提供される Intel ifort コンパイラの実行時ライブラリをリンクする必要があるかもしれません。

### 3.1.1. C または C++ からのライブラリの呼び出し

本ライブラリは C または C++ 環境からでもご利用いただけます。

ご利用の支援として Fortran と C の間の型マッピング情報を持った C/C++ ヘッダーファイル [INSTALL\_DIR]/c\_headers/nagmk25.h が提供されます。ヘッダーファイルから必要な部分だけを（ファイルの先頭にある #defines など忘れずに）自身のプログラムにコピー&ペーストするか、もしくはヘッダーファイルを単純にインクルードしてご利用ください。

C または C++ から NAG Fortran Library を呼び出す際のアドバイスは、ドキュメント [INSTALL\_DIR]/c\_headers/techdoc.html をご参照ください。

### 3.2. インターフェースブロック

NAG Fortran Library インターフェースブロック（引用仕様宣言）はライブラリルーチンの型と引数を定義します。Fortran プログラムからライブラリルーチン呼び出す際に必ず必要という性質のものではありませんが（ただし本製品で提供される Example を利用するには必要となります）、これを用いることでライブラリルーチンが正しく呼び出されているかどうかのチェックを Fortran コンパイラに任せる事ができます。具体的にはコンパイラが以下のチェックを行うことを可能とします。

- (a) サブルーチン呼び出しの整合性
- (b) 関数宣言の型
- (c) 引数の数
- (d) 引数の型

NAG Fortran Library インターフェースブロックファイルはチャプター毎のモジュールとして提供されますが、これらをまとめて一つにしたモジュールが提供されます。

nag\_library

これらのモジュールは、Intel Fortran コンパイラ (ifort) を用いて、プリコンパイルされた形式 (\*.mod ファイル) で提供されます。

コンパイル時に、-I"pathname" オプションを用いて、モジュールファイルが置かれているディレクトリのパス ([INSTALL\_DIR]/nag\_interface\_blocks) を指定してください。

提供されるモジュールファイル (.mod ファイル) はインストールノート「2.2. 開発環境」にあるコンパイラを用いて生成されています。モジュールファイルはコンパイラ依存のファイルであるため、ご利用のコンパイラとの間に互換性がない場合は、ご利用のコンパイラでモジュールファイルを生成する必要があります。（自身のプログラムでインターフェースブロックをご利用にならないのであれば、この限りではありません。ただし、Example プログラムはインターフェースブロックを利用しますので、Example プログラムをご利用になる場合は必要です。）



提供されるスクリプト `nag_recompile_mods` を用いて、モジュールファイルのセットを指定のディレクトリ（例えば `nag_interface_blocks_alt`）に生成することができます。

例)

```
[INSTALL_DIR]/scripts/nag_recompile_mods nag_interface_blocks_alt
```

このスクリプトは `PATH` 環境変数に設定された Intel Fortran コンパイラを使用します。

`[INSTALL_DIR]/scripts/nag_recompile_mods` を実行する前に、ご利用の Intel Fortran コンパイラの環境設定スクリプトを実行しておくとう安心です。

新しいモジュールファイルのセットをデフォルトのセットとして使用するには、元のモジュールファイルのセットを含むディレクトリ `[INSTALL_DIR]/nag_interface_block` の名前を `[INSTALL_DIR]/nag_interface_blocks_original` に変更し、新しいモジュールファイルのセットを含むディレクトリ `[INSTALL_DIR]/nag_interface_blocks_alt` の名前を `[INSTALL_DIR]/nag_interface_blocks` に変更してください。

### 3.3. Example プログラム

提供される Example 結果は、インストールノートの「2.2. 開発環境」に記載されている環境で生成されています。Example プログラムの実行結果は、異なる環境下（例えば、異なる Fortran コンパイラ、異なるコンパイラライブラリ、異なる BLAS または LAPACK ルーチンなど）で若干異なる場合があります。そのような違いが顕著な計算結果としては、固有ベクトル（スカラー（多くの場合 -1 ）倍の違い）、反復回数や関数評価、残差（その他マシン精度と同じくらい小さい量）などがあげられます。

提供される Example 結果は NAG スタティックライブラリ `libnag_mkl.a`（MKL 提供の BLAS / LAPACK ルーチンを使用）を用いて算出されています。NAG 提供の BLAS / LAPACK ルーチンを使用した場合、結果がわずかに異なるかもしれません。

Example プログラムは本ライブラリが想定する動作環境に適した状態で提供されます。そのため、ライブラリマニュアルに記載／提供されている Example プログラムに比べて、その内容が若干異なる場合があります。

`[INSTALL_DIR]/scripts` には `nag_example_mkl`, `nag_example_shar_mkl`, `nag_example`, `nag_example_shar` の 4 つのスク립トが提供されます。

これらのスク립トを用いて Example プログラムを簡単に利用する事ができます。

- `nag_example_mkl`  
NAG スタティックライブラリ `libnag_mkl.a` および本製品で提供される MKL ライブラリをリンクします。
- `nag_example_shar_mkl`  
NAG 共有ライブラリ `libnag_mkl.so` および本製品で提供される MKL ライブラリをリンクします。
- `nag_example`  
NAG スタティックライブラリ `libnag_nag.a` をリンクします。
- `nag_example_shar`  
NAG 共有ライブラリ `libnag_nag.so` をリンクします。

これらのスク립トは、Example プログラムのソースファイル（必要に応じて、データファイル、オプションファイルその他）をカレントディレクトリにコピーして、コンパイル／リンク／実行を行います。

ご利用の NAG ライブラリルーチンの名前をスクリプトの引数に指定してください.

例)

```
nag_example_mkl e04nrf
```

この例では, e04nrfe.f (ソースファイル), e04nrfe.d (データファイル), e04nrfe.opt (オプションファイル) をカレントディレクトリにコピーして, コンパイル／リンク／実行を行い e04nrfe.r (結果ファイル) を生成します.

### 3.4. Fortran 型と強調斜体文字の解釈

本ライブラリは 32-bit 整数を使用します.

ライブラリとライブラリマニュアルでは浮動小数点変数を以下のようにパラメーター化された型を用いて記述しています.

REAL (KIND=nag\_wp)

ここで nag\_wp は Fortran の種別パラメーターを表しています.

nag\_wp の値は製品毎に異なり, その値は nag\_library モジュールに定義されています.

これに加え, いくつかのルーチンで以下の型が使用されます.

REAL (KIND=nag\_rp)

これらの型の使用例については, 各種 Example プログラムをご参照ください.

本ライブラリでは, これらの型は次のような意味を持っています.

REAL (kind=nag\_rp)     - REAL (単精度実数)  
REAL (kind=nag\_wp)     - DOUBLE PRECISION (倍精度実数)  
COMPLEX (kind=nag\_rp) - COMPLEX (単精度複素数)  
COMPLEX (kind=nag\_wp) - 倍精度複素数 (e. g. COMPLEX\*16)

上記に加え, ライブラリマニュアルでは強調斜体文字を用いていくつかの用語を表現しています. 詳細は Essential Introduction の「4.4 実装依存情報」をご参照ください.

### 3.5. メンテナンスレベル

ライブラリのメンテナンスレベルは、ライブラリルーチン A00AAF の Example プログラムをコンパイル／リンク／実行することにより確認することができます。この時、スクリプト nag\_example\* を引数 a00aaf と共に用いれば、Example プログラムのコンパイル／リンク／実行を容易に行うことができます（「3.3. Example プログラム」参照）。ライブラリルーチン A00AAF はライブラリの詳細（タイトル、製品コード、使用されるコンパイラおよび精度、バージョン（Mark）など）を出力します。

#### 4. ルーチン固有の情報

本ライブラリルーチン固有の情報を（チャプター毎に）以下に示します.

##### a. F06, F07, F08, F16

多くの LAPACK ルーチンは “workspace query” メカニズムを利用します. ルーチン呼び出し側にどれだけのワークスペースが必要であるかを問い合わせるメカニズムですが, NAG 提供の LAPACK と MKL 提供の LAPACK ではこのワークスペースサイズが異なる場合がありますので注意してください.

libnag\_mkl.a, libnag\_mkl.so では, ベンダー提供の BLAS/LAPACK の問題を回避するために, 以下の BLAS/LAPACK ルーチンについては NAG 提供のものが含まれています（呼び出されます）.

dgeqrt	zgeqrt	dgemqrt	zgemqrt	dtpqrt
ztpqrt	dtpmqrt	ztpmqrt	dormrz	zunmrz

##### b. S07 – S21

これらのチャプターの関数の動作は, ライブラリ実装毎に異なります.

一般的な詳細はライブラリマニュアルをご参照ください.

本ライブラリ固有の値を以下に示します.

S07AAF  $F_1 = 1.0E+13$

$F_2 = 1.0E-14$

S10AAF  $E_1 = 1.8715E+1$

S10ABF  $E_1 = 7.080E+2$

S10ACF  $E_1 = 7.080E+2$

S13AAF  $x_{hi} = 7.083E+2$

S13ACF  $x_{hi} = 1.0E+16$

S13ADF  $x_{hi} = 1.0E+17$

S14AAF IFAIL = 1 if  $X > 1.70E+2$   
 IFAIL = 2 if  $X < -1.70E+2$   
 IFAIL = 3 if  $\text{abs}(X) < 2.23E-308$   
 S14ABF IFAIL = 2 if  $X > x\_big = 2.55E+305$

S15ADF  $x\_hi = 2.65E+1$   
 S15AEF  $x\_hi = 2.65E+1$   
 S15AGF IFAIL = 1 if  $X \geq 2.53E+307$   
 IFAIL = 2 if  $4.74E+7 \leq X < 2.53E+307$   
 IFAIL = 3 if  $X < -2.66E+1$

S17ACF IFAIL = 1 if  $X > 1.0E+16$   
 S17ADF IFAIL = 1 if  $X > 1.0E+16$   
 IFAIL = 3 if  $0 < X \leq 2.23E-308$   
 S17AEF IFAIL = 1 if  $\text{abs}(X) > 1.0E+16$   
 S17AFF IFAIL = 1 if  $\text{abs}(X) > 1.0E+16$   
 S17AGF IFAIL = 1 if  $X > 1.038E+2$   
 IFAIL = 2 if  $X < -5.7E+10$   
 S17AHF IFAIL = 1 if  $X > 1.041E+2$   
 IFAIL = 2 if  $X < -5.7E+10$   
 S17AJF IFAIL = 1 if  $X > 1.041E+2$   
 IFAIL = 2 if  $X < -1.9E+9$   
 S17AKF IFAIL = 1 if  $X > 1.041E+2$   
 IFAIL = 2 if  $X < -1.9E+9$

S17DCF IFAIL = 2 if  $\text{abs}(Z) < 3.92223E-305$   
 IFAIL = 4 if  $\text{abs}(Z) \text{ or } FNU+N-1 > 3.27679E+4$   
 IFAIL = 5 if  $\text{abs}(Z) \text{ or } FNU+N-1 > 1.07374E+9$   
 S17DEF IFAIL = 2 if  $\text{AIMAG}(Z) > 7.00921E+2$   
 IFAIL = 3 if  $\text{abs}(Z) \text{ or } FNU+N-1 > 3.27679E+4$   
 IFAIL = 4 if  $\text{abs}(Z) \text{ or } FNU+N-1 > 1.07374E+9$   
 S17DGF IFAIL = 3 if  $\text{abs}(Z) > 1.02399E+3$   
 IFAIL = 4 if  $\text{abs}(Z) > 1.04857E+6$   
 S17DHF IFAIL = 3 if  $\text{abs}(Z) > 1.02399E+3$   
 IFAIL = 4 if  $\text{abs}(Z) > 1.04857E+6$   
 S17DLF IFAIL = 2 if  $\text{abs}(Z) < 3.92223E-305$   
 IFAIL = 4 if  $\text{abs}(Z) \text{ or } FNU+N-1 > 3.27679E+4$

IFAIL = 5 if  $\text{abs}(Z)$  or  $\text{FNU}+\text{N}-1 > 1.07374\text{E}+9$

S18ADF IFAIL = 2 if  $0 < X \leq 2.23\text{E}-308$

S18AEF IFAIL = 1 if  $\text{abs}(X) > 7.116\text{E}+2$

S18AFF IFAIL = 1 if  $\text{abs}(X) > 7.116\text{E}+2$

S18DCF IFAIL = 2 if  $\text{abs}(Z) < 3.92223\text{E}-305$   
 IFAIL = 4 if  $\text{abs}(Z)$  or  $\text{FNU}+\text{N}-1 > 3.27679\text{E}+4$   
 IFAIL = 5 if  $\text{abs}(Z)$  or  $\text{FNU}+\text{N}-1 > 1.07374\text{E}+9$

S18DEF IFAIL = 2 if  $\text{REAL}(Z) > 7.00921\text{E}+2$   
 IFAIL = 3 if  $\text{abs}(Z)$  or  $\text{FNU}+\text{N}-1 > 3.27679\text{E}+4$   
 IFAIL = 4 if  $\text{abs}(Z)$  or  $\text{FNU}+\text{N}-1 > 1.07374\text{E}+9$

S19AAF IFAIL = 1 if  $\text{abs}(X) \geq 5.04818\text{E}+1$

S19ABF IFAIL = 1 if  $\text{abs}(X) \geq 5.04818\text{E}+1$

S19ACF IFAIL = 1 if  $X > 9.9726\text{E}+2$

S19ADF IFAIL = 1 if  $X > 9.9726\text{E}+2$

S21BCF IFAIL = 3 if an argument  $< 1.583\text{E}-205$   
 IFAIL = 4 if an argument  $\geq 3.765\text{E}+202$

S21BDF IFAIL = 3 if an argument  $< 2.813\text{E}-103$   
 IFAIL = 4 if an argument  $\geq 1.407\text{E}+102$



**c. X01**

数学定数は以下のとおりです.

X01AAF (pi) = 3.1415926535897932

X01ABF (gamma) = 0.5772156649015328

**d. X02**

マシン定数は以下のとおりです.

浮動小数点演算の基本的なパラメーター :

X02BHF = 2

X02BJF = 53

X02BKF = -1021

X02BLF = 1024

浮動小数点演算の派生的なパラメーター :

X02AJF = 1.11022302462516E-16

X02AKF = 2.22507385850721E-308

X02ALF = 1.79769313486231E+308

X02AMF = 2.22507385850721E-308

X02ANF = 2.22507385850721E-308

コンピューター環境のその他のパラメーター :

X02AHF = 1.42724769270596E+45

X02BBF = 2147483647

X02BEF = 15

**e. X04**

エラーメッセージおよびアドバイスメッセージのデフォルトの出力先装置番号は 6 番となります.

## 5. ドキュメント

ライブラリマニュアルは本製品の一部として提供されます。

また、NAG のウェブサイトからダウンロードすることもできます。

ライブラリマニュアルの最新版は以下のウェブサイトをご参照ください。

<http://www.nag.co.uk/numeric/FL/FLdocumentation.asp>

ライブラリマニュアルは以下の形式で提供されます。

- HTML5 - HTML/MathML マニュアル（各ドキュメントの PDF 版へのリンクを含む）
- PDF - PDF マニュアル（PDF のしおり、または HTML 目次ファイルから閲覧する）

これらの形式に対して、以下の目次ファイルが提供されます。

nagdoc\_fl25/html/frontmatter/manconts.html

nagdoc\_fl25/pdf/frontmatter/manconts.pdf

nagdoc\_fl25/pdf/frontmatter/manconts.html

また、これらの目次ファイルへのリンクをまとめたマスター目次ファイルが提供されます。

nagdoc\_fl25/index.html

各形式の閲覧方法および操作方法については、ライブラリマニュアルの“Online Documentation”ドキュメントをご参照ください。

加えて、以下のドキュメントが提供されます。

- in.html - インストールノート（英語版）
- un.html - ユーザーノート（英語版）

## 6. サポート

製品のご利用に関してご質問等がございましたら、電子メールにて「日本 NAG ヘルプデスク」までお問い合わせください。その際、ご利用の製品の製品コード（FLL6I25DCL）とお客様の保守 ID をご明記いただきますようお願い致します。ご返答は平日 9:30～12:00, 13:00～17:30 に行わせていただきます。何卒よろしくお願い致します。

日本 NAG ヘルプデスク

Email: [naghelp@nag-j.co.jp](mailto:naghelp@nag-j.co.jp)

## 7. コンタクト情報

日本ニューメリカルアルゴリズムズグループ株式会社（日本 NAG）

〒104-0032

東京都中央区八丁堀 4-9-9 八丁堀フロンティアビル 2F

Email: [sales@nag-j.co.jp](mailto:sales@nag-j.co.jp)

Tel: 03-5542-6311

Fax: 03-5542-6312

NAG のウェブサイトでは製品およびサービスに関する情報を定期的に更新しています。

<http://www.nag-j.co.jp/> （日本）

<http://www.nag.co.uk/> （英国本社）

<http://www.nag.com/> （米国）