アンケート | 絶滅危惧種Fortranのために

【アンケートの主旨】
科学技術計算で広く使われて来たFortranですが、大学の講義で教えられることもなくなり、年々、利用者は減っていると思われます。「絶滅危惧種」とも揶揄されますが、大規模シミュレーションでは、今も多くのユーザーが利用しているのではないでしょうか(かくいう私もその一人です)。このアンケート調査は、数値計算でプログラミング言語を利用するユーザーの声を集め、Fortranの現状と展望を語るための貴重な意見として役立てたい、というものです。いただいた回答は統計データとして、「並列Fortranに関するシンポジウム」
http://site.hpfpc.org/home/events/parallel_fortran_sympo
で紹介させていただきます。
ご趣旨を理解いただき、是非、よろしくご協力の程、お願いいたします。

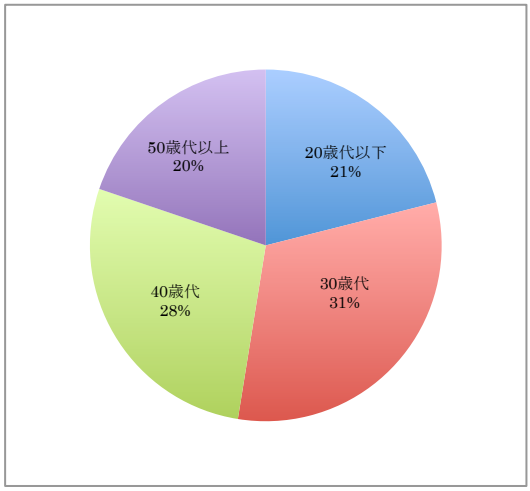
名大・理・渡邉智彦
アンケート回答期限：2015-08-12

上記のアンケートを、物理学会領域 2、天文学会、地球電磁気・地球惑星圏学会、プラズマ・核融合学会のメーリングリストにて呼びかけたところ、1 週間で 394 件もの回答をいただきました。皆様の熱い思いとご協力に改めて感謝申し上げます。
（渡邉智彦; 2015 年 9 月 2 日）

【アンケート結果のまとめ】

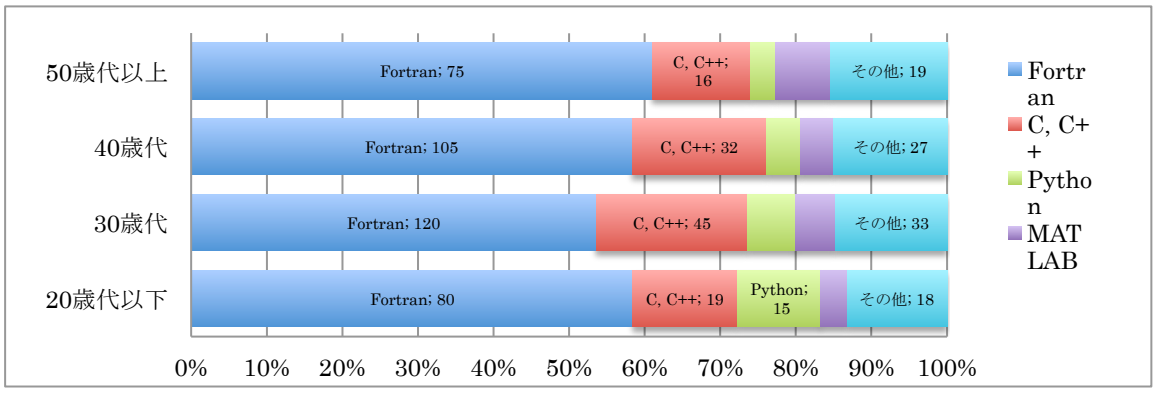
Q1: 年齢層別のデータを得るため、あなたの年代を教えてください。

20 歳代以下	30 歳代	40 歳代	50 歳代以上
83	124	109	78



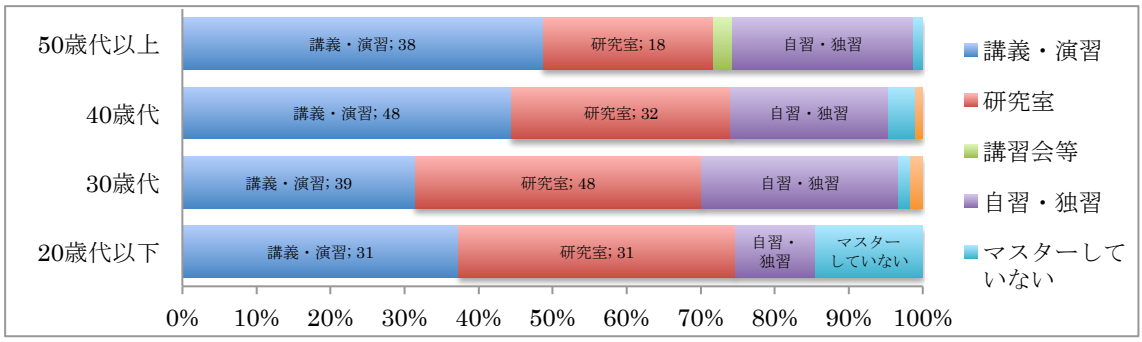
Q2: 科学計算にどんなプログラミング言語を使いますか?(複数回答可)

	Fortran	C, C++	Python	MATLAB	その他
20 歳代以下	80	19	15	5	18
30 歳代	120	45	14	12	33
40 歳代	105	32	8	8	27
50 歳代以上	75	16	4	9	19



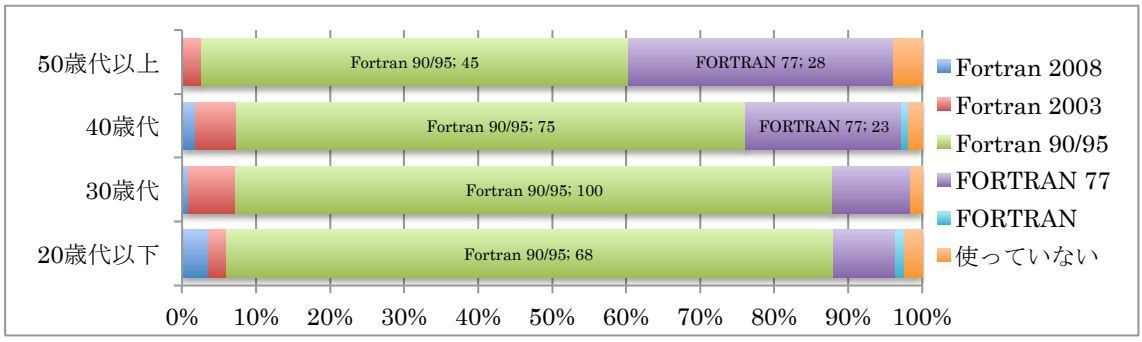
Q3: Fortran をどこでマスターしましたか?

	講義・演習	研究室	講習会等	自習・独習	マスターしていない	その他
20 歳代以下	31	31	0	9	12	0
30 歳代	39	48	0	33	2	2
40 歳代	48	32	0	23	4	1
50 歳代以上	38	18	2	19	1	0



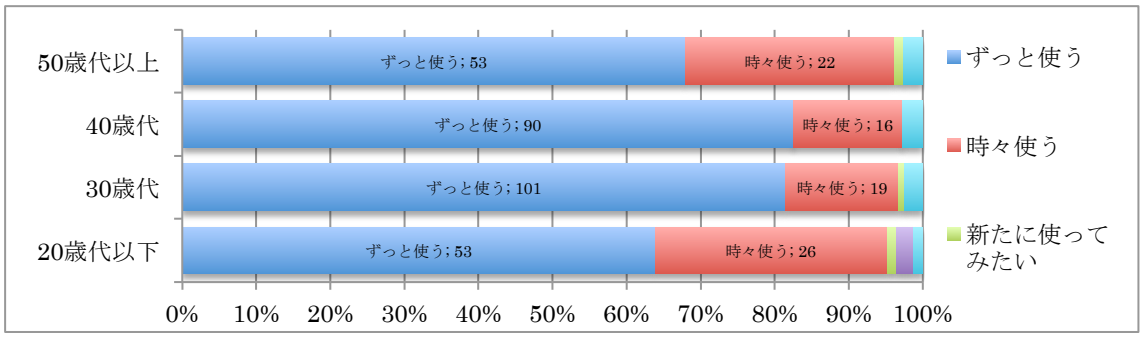
Q4: Fortran はどのバージョンを使っていますか?

	Fortran 2008	Fortran 2003	Fortran 90/95	FORTRAN 77	FORTRAN	使っていない
20 歳代以下	3	2	68	7	1	2
30 歳代	1	8	100	13	0	2
40 歳代	2	6	75	23	1	2
50 歳代以上	0	2	45	28	0	3



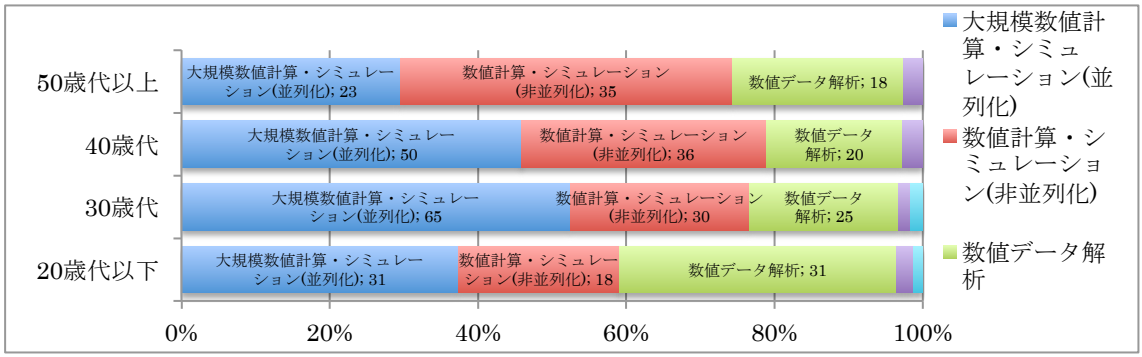
Q5: Fortran を今後も使いますか?

	ずっと使う	時々使う	新たに使ってみたい	他の言語に乗り換えるつもり	今後も使わない
20 歳代以下	53	26	1	2	1
30 歳代	101	19	1	0	3
40 歳代	90	16	0	0	3
50 歳代以上	53	22	1	0	2



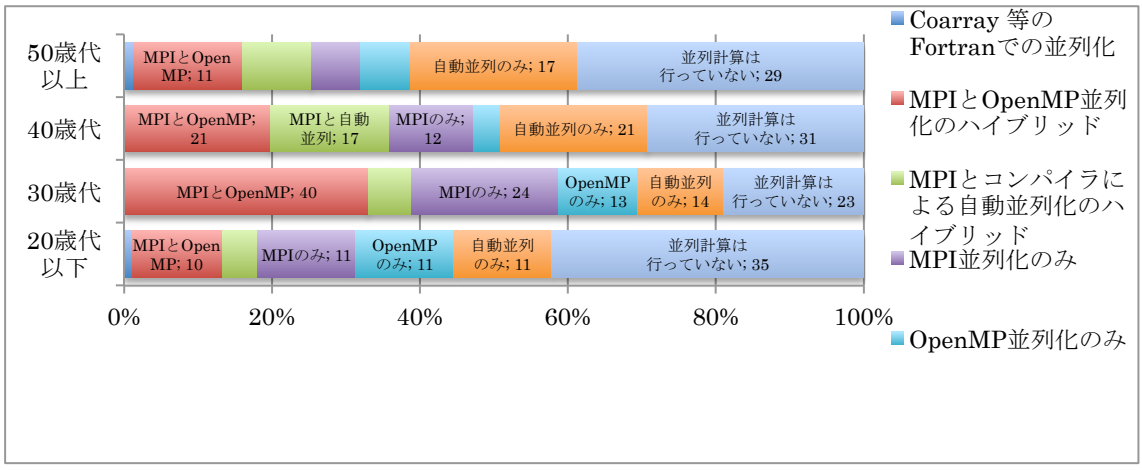
Q6: Fortran を使う用途は何ですか?

	大規模数値計算・シミュレーション(並列化)	数値計算・シミュレーション(非並列化)	数値データ解析	使わない	その他
20 歳代以下	31	18	31	2	1
30 歳代	65	30	25	2	2
40 歳代	50	36	20	3	0
50 歳代以上	23	35	18	2	0



Q7: Fortran では主にどのような並列化を行っていますか?

	Coarray 等の Fortran での並列化	MPI と OpenMP 並列化のハイブリッド	MPI とコンパイラによる自動並列化のハイブリッド	MPI 並列化のみ	OpenMP 並列化のみ	コンパイラによる自動並列化のみ	並列計算は行っていない
20 歳代以下	1	10	4	11	11	11	35
30 歳代	0	40	7	24	13	14	23
40 歳代	0	21	17	12	4	21	31
50 歳代以上	1	11	7	5	5	17	29



Q8: Fortran を使う理由、または使わない理由を教えてください。

- 一番最初に覚えた言語だから。コンパイラによる最適化が良く働き、高速。複素倍精度変数とその演算や関数が標準で実装されている。
- 細かな機能面で C に劣る部分もあるが、その分記述や構造がシンプルであるため。
- 数値シミュレーション用のプログラムを作る上で必要十分な機能を備えているから。また、C を使わないので実際がどうかは不明だが、FORTRAN の方が大規模高速演算のためのコンパイラの最適化が進んでいると聞くので。
- 継承してきたコードが Fortran である
  - 数式の記述が簡単
  - 大抵のマシンには最適化コンパイラが存在している
- Fortran のみ知っているので
- これまで Fortran で開発されてきたコードを拡張して用いているため。
- 科学計算(私の専門分野)においては、Fortran が広く用いられており、必要であるから。
- これまでに書かれたソースコードの活用。文字列処理が楽
- 使い慣れているから
- 使う理由
  - 先人が作成した多くのライブラリが存在している
  - 数式の実装が直感的に出来るため、思考が妨げられない
  - メモリに関する(余計な)知識がなくても実装が可能である
  - これまでのコードがほぼ全て Fortran 実装であるため
  - 慣れている
- 平易な言語であり書きやすいから.また,現時点で他の言語に移る必要性を感じていないから.
- 計算のための言語である事と、何より慣れ親しんでいることから
- 使う理由:過去数十年の資産、他人から頂くコードはたいてい fortran である。自分で 0 からプログラムを書き始める時に(C 言語と比べて)数式に近い記述が可能で負担が少ない。同様にファイル入出力の記述についても負担が少ない。受け持つことになったプログラミング講義では fortran を教えることになっている。
- これまで作ってきたプログラムが FORTRAN であり,今更変更はできないので
- 使いやすい。分かりやすい。interface しやすい。バグしにくい。
- 配列演算のプログラミングが便利。
- 計算機シミュレーションに適しているから。
- 過去からのライブラリーが充実しているから

- 使う理由:業界(理論天文学)で広く使われているから。
- Fortran で書かれているコードを使う時のみ使うが、C バージョンがあれば使わない
  - 直観的なプログラミングが可能
  - 複素数を扱いやすい
- なじみがあるから。
- 使う理由:
  - 数値計算に特化した言語であり、高速かつ使い勝手が良い。
- 使わない理由:
  - C/C++のように多様な形式のファイル IO を想定しておらず、画像処理などには使いづらい。
- 使い慣れたアルゴリズムであり、自分で作ったツールがそろっているため。
- 過去の遺産を引き継ぐため、標準電離圏モデル、大気モデル、イオノグラム実変換など多くのモデルが Fortran で書かれ、改良されてきた。殆どのデータ処理(ビット演算なども)は Fortran でも可能なので、大きな理由は「いまさら他の言語も必要としない」でしょうか。
- メイン研究でのシミュレーションで使用
- それしか知らないから。
- 宇宙科学系でスタンダードに使われているため。
  配列処理が楽。
- もう歳なので、今更別の言語を覚えたくない
  F77 は要素で記述できるのでプログラムには安心感がある
  IDL, f90 の行列計算は怖い
- 過去のプログラムの蓄積。慣れた言語。動作の安定性。
  - 慣れていて使いやすいから。
  - 他の多くの研究者も Fortran を使っており,数値計算に関し,彼らとの情報交換がしやすいから。
  - 特に大型計算機では,Fortran コンパイラの方が性能がよいと感じるから。
- 使わない理由:必要性を感じない。
- オリジナルの C 言語に比べて数値計算に向いていること、また、過去に開発されたプログラムの多くが fortran でかかれており、それを利用するには fortran をメインで使う方が効率的であること。
- 慣れ親しんだものの方が、ミスが少ないと考えられるから。
- 数値計算のコードが書き易いため

- 既存のデータベースや書籍が豊富で使いやすい.
  - プログラム自体が書きやすい.
- 複素数が扱いやすい
  構造がしっかりしている
- 実言語に近い感覚で、プログラムが書けるから。
  F90 以後、格段に使いやすくなったから。
- これまで書いていたプログラムが Fortran だから。
  他の言語をあまり知らないから。
- c++のコンパイラの最適化機能が fortran コンパイラに劣るため。
- 適度な機能、体に染みついている
- これまでの資産の活用。
- 使う理由慣れと速さ.使わないのは文字処理のとき.
- 使う理由:物理過程の数値シミュレーションに必要十分な機能がシンプルにまとまっているから。私は物理学の専門家であり,プログラミングは道具として使っているに過ぎない。余計な機能がないほうが,必要な作業に集中できる。
- 自分のコードの多くが Fortran で書かれているので、特に他の言語に移植する理由がない限り、使い続ける。
- 不要な API や命令が無い・少ない・あまり気にしないでよいのでコードに集中できる。
  - 文字列処理については少なくとも C 系列より優れる。
  - 世界的な規格化がすすんでいること、DEC 以来市場で優越したベンダが存在しないため、方言がなく、可搬性に優れる。
- C のセミコロンが面倒
  C の配列の確保が面倒
- 情報分野でメジャーな言語でない
  開発環境として優れたものが思い浮かばない
- 大型計算機で最適化されている(はずと思っている)から
  オブジェクト指向を必要としていない
- 文法が分かりやすく、初学者でも使いこなしやすいと考えるため
- シンプルで使いやすく, 科学的な数値計算に向いていると思うから.
- 過去に蓄積されたコード資産を利用できるのが、Fortran を使う理由です。
  一方、Fortran 2003/2008 の(特に日本語での)情報はまだ乏しいですね。

- fortran は個人的にコードがすっきりしていて見やすいような気がするので純粋に数値計算のみの用途の場合は fortran を使う。ただしバイナリデータの扱いが他のソフトウェアとの連携を考えるときに不便だと思う。
- This is the only language I can use.　I learned C/C++ later, but it is not so easy to use a different language.　Even FORTRAN 95 is difficult to use for me (for unfamiliarity).　Ah.. I work abroad, and never installed Japanese generating soft to my machine.
- 先輩方から受け継いだ過去の遺産がほとんど Fortran なのと、自分が一番理解している言語が Fortran であるため、計算のためのプログラミングに使うのは Fortran がほとんど。
- 必要な場面が無く、新たに勉強する暇がない
- 慣れているので。
  - 学部時代からずっと使っているから
  - 天文関連の公開コードで利用されているから
- フリーで手に入る各種ライブラリが豊富
- 使い慣れているので部分的に(高速化が必要なところだけ)使う。
- 余計な機能がなく、シンプルで使いやすい。その結果コンパイラも最適化を行いやすい。最近の C では可能だが、複数次元の配列が Fortran は扱いやすい。
- 科学技術計算用の計算機言語としてはもっとも習熟している言語と思うから使っている
- 信頼性。
- 大規模計算機上で使用できるコンパイラがほぼ Fortran と C 言語であるため。
  今までに開発された Fortran のプログラム資源があるため。
  Fortran におけるコーディング・チューニング技法に慣れているため。
- 慣れている
  他はほとんど使ったことがない
- 既存の大規模計算コードが全て Fortran で書かれているため
- 配列計算が簡単に書ける。関数が配列を返せる。
- Fortran で私が計算機でやりたいことは何でもできるから
- これまで作られたプログラムの豊富な蓄積。
  慣れ親しんでいる。
  C 言語も使っているが、科学技術計算では特に便利ということはない。
  改良の余地は多々あるが、科学技術計算プログラムの入門への教育、特にプログラム作成の論理的な思考法を習得させるには Fortran が適している。数式との整合性が良く、とっつきやすい。

大学の講義で Fortran を教えて、使っている。

- 過去の蓄積
- ライブラリーが充実している。
  過去の遺産(自分、他人ともに)がある。
  慣れている。文法も熟知している。
- 他の言語は本格的には使えないため。
  プログラムがコンパクトにかける。
- 共同研究上、C や C++を使うこともありますが、自分が主導的に書く場合は、Fortran を使います。主な理由は、パフォーマンスが高く、さらには、バグが少ないコードが書けるからです。例えば、C++でコードを書くと、どんなにデバッギングをしっかり行っても、メモリーリークなどのトラブルは必ず起こります。
- 計算速度が速い
- 他の言語より定義が明快で、エラーを見つけやすい(一応、Maylab, IDL もたしなんでいる、C は入力エラーの際の影響が大きすぎて止めた)
- 過去に Fortran で作ったプログラムがあり、それを活用しているため。
- 使う理由
  - 組込み関数が直感的で豊富.
  - 動的配列確保の手軽さ.
  - 少々粗い PG でも計算精度が維持されていることが多い.
- C は使い慣れていないので何かと面倒
- これまでのリソースの豊富さとコーディングおよび判読性
- 昔から使っております。
  また、小生はそもそも実験屋なので、理論計算にあらたに長い時間を割くことには抵抗があります。あくまで、実験結果を考察するために行うレベルの計算ですので。
- 知っている唯一の言語ですので。
- 研究開始当初は使っていたが、元々子供時代から C++(Visual Studio)を使い続けていたこともあり、Fortran の言語使用には苛立ちを感じていた。学生を指導するようになり、彼らが就職先で活躍する手段を身に着けさせる意味で、C++に移行した。現在の企業では、Fortran を使えると言っても、開発現場に回して貰えない。
- プラズマ物理関連で私が使用しているコードの多くが fortran で記述されており、コード資産の引き継ぎの事を考えると、しばらくは使い続けると思います。(逆に言うと、世界中のプラズマ物理研究者の多くが他の言語を使うようになったら乗り換えると思います)

- 旧来の資産があるため。また、コンパイラによる最適化・自動ベクトル化・並列化の恩恵を得やすいため。
- 慣れているから
- 元々の数値計算コードが Fortran だったため現在も利用している
- 過去のプログラムを C に書き換える時間がないので、Fortran のまま使用している。
- Fortran を使う理由:
  配列演算用の簡便な記法が利用可能
  コンパイラによる最適化が効きやすい
- 数値計算のアルゴリズムの実装がしやすい。
  数値演算の蓄積が多い。
- 研究のために。生きるために。
- Fortran 以外に知らないから。
- 私は理論屋であり数値計算のスペシャリストではないため、一度コードを組むと別の言語に書き換えるのが億劫であるため
- これまでのプログラムの蓄積を生かすために利用。
- 研究分野の有名な公開コードが Fortran で書かれていたため。
  もし、他言語で書かれていたらおそらく Fortran を覚えることはなかったと思う。
- プログラミングにおいて、計算部分の記述に集中できる。
- 使っている理由は、宇宙業界で慣習となっているということと、シンプルな言語であるため覚えやすかったことである。
- 大学院入学時からずっと使っており馴染みがある。高級言語のためコーディングしやすい。
- fortran 全盛期に作ったコードを今でも使っている
- 自分にとって一番わかりやすく、なじみのある言語だから。
  ライブラリーなど、過去の遺産が充実していて便利だから。
- 直感的に書きやすいから。
- 新しい言語を使う必要性を感じない。
- 過去のプログラムの蓄積がある。シンプルである。
- 自動最適化が容易であるため
- 数値計算する上で必要十分の機能を持っているため。
  他の用途では C 言語を使っているが、数値計算で C のメリットは見当たらない。
- Fortran を使う理由: これまでずっと使ってきて慣れているから.
- Library

- 過去のプログラムの遺産に Fortran code があり、重要である為という理由が最も大きいのが実情。
- 他の言語を使ったことがないため。また、周囲の研究者との共通言語となっているので便利。
- 過去のコードの蓄積がある。
  数式表現がわかりやすい。
- 配列の扱いが直観的に分かりやすい。
  C などと比べて環境に依存する予期せぬエラーが起こりにくい。
- 直感的にコードを書くことができ、C に比べてストレスを感じなかったからです。
- これ以外の言語には余り詳しくないので使っている。
- Fortran で行われたシミュレーションデータの提供を受けて、現在の研究活動を行っているため。
- 変数構造・宣言などの文法が数値計算に向いているから。
  プログラムを解読し易いから。
- これまでの資産(ソースコード、教科書)を利用し、効率よく研究を進める為に Fortran を使う。
- 既存のプログラムが Fortran で作られているうえ、他の言語に移植するには相当な労力と費用がかかるため。
- これまでの資産を継承するため
- 新しい言語を学ぶのがおっくうだから
- 数値計算をするのに十分な機能がある。特に、ベクトル計算の記述が簡易で良い。
  - 他の言語は、数値計算に必要の無い機能がある。例えば、手続き型プログラミング、抽象化、オブジェクト指向、ジェネリックプログラミングのサポートは不用である。
- 科学計算のサブルーチンがそろっている。
- 学生の頃研究室で教えられて以来ずっと使ってきたが、特にほかに乗り換える積極的な理由がないため。処理が速く、大規模計算に適していると聞いているため。
- 昔から使っているし使い慣れているから
  - 過去の資産の活用
  - 数値計算には Fortran がシンプル
- 従来の機能で十分だと思うので。
- 既存コードが FORTRAN で提供されるので
- 使う理由:数値計算をよくやるので(他の言語では面倒な事が多い)。近年、市販の回路シミュレータや電磁場計算ソフトウェアが多く出回っているが、計算対象モデルがちょっ

と複雑だと計算できない、ということが多くなってきている。特に、最近は、電磁気・相対論・熱・回路・剛体等の複合要素を合わせた計算を行うことが多く、自前でプログラムを組む必要がある。

- 配列計算が重くなりそうな場合，ループが多い場合に使用する.
  計算が軽い場合は package の導入がしやすいものを使うため，使わない.
- 使い慣れているから
- 科学計算にはもっとも適しているから
- 使う理由。比較的習熟しているため。過去に開発されたコードの活用のため。
- 数値計算に適している。
  多くのサブルーチンライブラリがある。
  昔から使っていて慣れている。
- 現在の Fortran77 の機能で用が足りている。また、新たにプログラム言語を覚えるのは面倒。
- 言語として多次元配列をサポートする唯一の言語
  - 配列への添字アクセスが数学的概念と一致し分かりやすい
  - 言語として並列化をサポートする唯一の言語
  - 常に改訂が行われ、処理速度に主眼がおかれながらも言語としての機能向上継続して行われており、最も近代的な言語である
- 分野で用いられている既存のコードの多くが Fortran で記述されているため.また,コンパイラによる最適化がされやすいとされているため.
- 研究者の間に未だに Fortran イコール 77 という根強い 77 ユーザーが多いことが、大学教育における Fortran 評価の極端な二分化を招いている。77 を教える時代遅れな授業と Fortran 自体を避ける授業への二分化である。77 の文化は現代的なプログラミングを教育する上での悪影響が大きい。関数が値渡しに対応していないことや、構造体が無いことなどである。
- それしか知らないから
- 慣れ親しんでいることや、構築性の高い言語であることを気に入っている。
- 大規模計算にチャレンジしたいから
- 作成・使用しているコードが Fortran で記述しているため、計算環境が消失するとお手上げになってしまう。
  - 数式や多次元配列を扱うには,C より便利だから.
  - 実行が速いから
  - 今までの資産があるから

- 科学計算に向いている。
- Fortran で書かれた既存の数値計算コードが多いため,今後も Fortran を使い続ける.
- 過去の遺産が充実している.慣れている.複素数が容易に扱える.
- 使いやすい言語だと思うから。
  他の言語をきちんと勉強していないから。
- これまで使ってきたということですが、少なくとも性能に不満はありません。(より高速にはありますが)
- 過去の program の蓄積、実験との比較で確認した信頼性。
- 資産があるので仕方が無く使っている。
- 学生時代より親しんだ言語であることと、今でも多くのシミュレーションコードがフォートランで記述されているため
- 現在では、使い慣れた言語(C/C++)を用いることが多い。研究テーマの一つで、共同研究先で開発された計算コードが Fortran であるため、研究テーマの中の一つでのみ使用している。データ処理や簡易な数値計算であれば、現在では Python を用いるのが便利なので、Fortran はほぼ使用しない。
- 過去の資産の利用と高速性のため。
  python と組み合わせても利用できるため、高速化が必要な場合に、利用を検討する。
- 複雑な物理モデリング、そのグラフィックス、データベース、ネットワークと連携のために、オブジェクト指向プログラミング言語に移行したため。
- 大規模数値計算をしないので、インタープリター型言語に乗り換えようとときどき思い立つも、過去の蓄積を考えると踏ん切りがつかず、現在に至る。
- スペクトル法を用いるとき,直交関数展開の公開サブルーチンが Fortran であるため
- 慣れ親しんだ言語であるので,Fotran なら確実に数値計算が行える自信があるから.
- 原始的なのでプログラマの言いなりであり、妙なことをおこさない。
- 数式演算のコーディングに便利だから.
- 数値モデルの開発および必要なライブラリが Fortran だから。
- 使う理由: 数値計算用途で書きやすいから
  使わない理由: データ解析で文字列処理をしたいときに困るから(標準化の範囲が狭く機能が貧弱である)
- FORTRAN で書かれたコードが職場に多くあるため
  FORTRAN は数値計算に用いるのが容易であるため
- 慣れているから

- 自身の研究や学生向けの演習には、C++ か Ruby を用いていますが、大規模計算機の C++ コンパイラの信頼性の低さと共同研究者の C++ への理解の無さから、fortran を今後も使わざるをえないかと。
- 分かりやすいから
- 数値計算コードのコーディングにおいて、c のように細かい手続きを考える必要が無い。数値計算用のプログラミングがとにかく楽で便利。
- 過去から使用している Fortran コードの蓄積がある.
- 他の言語と違い講義で教わった言語であるので,使いこなせる気がするため(他の言語は独習)。
- 書きやすいから。
- スーパーコンピューターを用いる大規模並列計算や、専門的な科学技術計算に広く用いられているから。
- 計算が速い。これまでの資産の継承。プログラミングも難しくない。
- 使う理由:
  do ループ・if 文など構造がイメージしやすく使いやすい。
  簡単なシミュレーションを行う際、0 からアルゴリズム構築がしやすい(言語が簡便であるため)。
- 使う理由:(1) 手軽にプログラムが書けるから。C より速くプログラムが書けます。(2) 複素数が扱えるから。C++ で複素数を扱うより fortran の方が楽です。
- 現在関わっている仕事で使っている共同開発のプログラムが Fortran で書かれているから。
- Fortran 以外の、安心して大規模計算に使える言語を知らないので。
- 昔は Fortran77 を用いていたが,最近は C 言語に乗り換えている。
  以前は,C 言語では複素数の計算ができなかったが最近はそれがなくなり,C 言語のデメリットが見られない。C 言語のメリットとして,構造体ならびに関数ポインタの利用が大きいが Fortran 77 ではできなかった。
  最近の Fortran は不明だが,C 言語にデメリットを感じないので,新しい言語を覚える気にはならないため,滅多に使わない。
  強いて使う場合は,library が Fortran 用として提供されている場合に限られる。
- スパコンでサポートされているから使用している。Intel コンパイラが利用できるため使用している。
- 既存のコードが全て Fortran。特に他の言語に書き換える必要がない。
- 並列化がしやすく,大規模計算に向いている.

- 分野的に fortran を使用するのが主流であるため。
  fortran 以外の言語を教えてくれる人がいない。
- これまでの数値計算や可視化のコード群が膨大に存在するために使っている。
- 気象分野の研究者です。Fortran は温暖化予測などの気候シミュレーションやデータ解析で日常的に使っています。Fortran は絶滅危惧種ではなく、後世に残すべき優れた原始言語だと思っています。現代人類はネオンデルタール人が用いていた言語が分かりませんが、分かったらどんなに知見が増えることでしょう。コンピュータ言語に、言語消滅はあってはならないと思います。
- 慣れているので使う。また、コード内の数式が見やすい。
- 使い慣れている。また、他の言語へ乗り換える積極的な理由がない。
  他の言語と比べて Fortran が劣っている理由を知らない。
- シミュレーションに用いる最先端の数値モデルが軒並み fortran 主体なので。
- 高速計算にやはり最適なのは未だに Fortran だと思う。長年のハウツーが蓄積されている。ライブラリの充実など(いくら C からも利用できるといっても)も捨てがたい。
- モデルコードがフォートランで書かれているから
- バイト単位でのデータ操作が行い易く,grads のデータを作成しやすい.
- 過去のプログラム資源を活かせるから。
  また、文法が容易であるため。
- 多次元配列を用いた高速なプログラムが、容易に実装できる点
- 自身の研究に必要だから。
- 私の研究分野のシミュレーションプログラムの多くが fortran で書かれているため。
- まわりがほとんど Fortran を使っているため。
  ずっと使ってきて慣れているということ以外に Fortran でなければならない理由はあまりないが、他に変える理由も特にない。
- 数値計算で学生が Fortran を使っているが,自分自身では一部の大規模なデータの解析・handling にのみ使っている.他の作業では Matlab の方が効率が高いため.
- 最も慣れ親しんでおり、速くプログラムが書けるため。
  - 豊富な数値計算ライブラリの存在.
- 師匠が Fortran 使いだったため、最初に習得したデータ解析手法であり、自分にとって最も使いやすいため
- 気象シミュレーションが主に Fortran で構成されている
  データの読み込みがしやすい
- 最初に学んだ言語であること。

ユーザーとして利用していうる大規模数値モデルが Fortran で記述されているため、一部改変する場合など必要不可欠であるため。
同じ研究分野の人もほぼ Fortran で仕事をしているため、Fortran で書かなければ、プログラムの供与等が円滑に行われないため。

- 数値計算に特化しているので、C よりも無駄が少なく効率的だから。たとえば、バグが出にくい、最適化しやすい、自由度が低いので移植性や交換性が高い、など。
- 計算速度。
  C よりかはマシな文法
- 他に知らないのが Fortran を使う最大の理由
- 表記が分かりやすい
- 慣れているのでコードを書きやすい。
  他の言語で書いた場合より処理が速いことが多い。
- 機能に不満はないので、他言語に乗り換える理由がない。
- 昔、大規模計算する時は Fortran の方は性能がでるという話を聞いたため。
- C のようになんでもかんでもポインタを使わず楽
- 使う理由:
  使い慣れているから。周囲も使っているから。
  計算することが目的であれば、分かりやすい言語だと思う。
- 過去の遺産が多い。
  コーディングが楽
  Fortran90 以降のバージョンはそれほど古めかしい言語という訳ではない。
- データ解析を行う際に、C などに比べて、処理速度が早いために利用している。
- これまで開発した資産を新しい言語に書き換える理由がないため。
- 既存のプログラムが fortran であるため使っている
- fortran が最も慣れた言語なので。
- Fortran は、他の言語に比べて、より人間の言葉に近く(機械語から遠く)、算術関数などの扱いも容易なので、科学計算にはとても便利であるので。また、他の言語をこれから習得するのは、なかなか難しいので。
- 一番初めに勉強したプログラミング言語であるため.
- 気象コードが Fortran で書かれているため
- これしか知らない。
- そこに Fortran があるから。

- 配列演算の効率的記法、チューニング度合いなどのメリットが大きい(大きかった)とおもいます。しかし、15 年程度前までのベクトル機 + Fortran のメリットは現在の intel CPU massive parallel 機ではほぼ消失したような印象もあります。Numerical Recipe が C++ になってしまいましたしねぇ。
- 過去の遺産があるので使用する。新たなプログラムを書く時は別の言語を使うことが多い。
- 大学院の研究で必要なため
- 使い慣れている上、仕事に使うプログラミング言語として十分な性能があるため。
  - 使う理由 1:よく使っている既存の数値計算プログラムやライブラリが Fortran 90/95 で書かれており、それを改変する形で利用することが多いため。
  - 使う理由 2:共同研究者が Fortran をメインで使っているため、Fortran で書いたほうがプログラムの融通がしやすい。
- Fortran は C 言語と比べて文字を打つ量が多いものの、プログラムコードが分かりやすく読みやすいため、使っている。
  Fortran を使えば、自分の意図するプログラムを組みやすくなるだけでなく、他人の作ったプログラムの解読もしやすくなる。
- 機会がない
- 速度を重要視しているため.
  マシンフレンドリー.
- 書き慣れている.速度を出しやすい.
- 自分のスキルとして Fortran しか自由にプログラムできないから。
- 書き馴れていること、計算速度を速くしやすいこと。
  過去の資源を再利用できること。
  配列の取り扱いが容易なこと。
- 使いやすい,分かり易い
- C 言語よりも、文法がわかりやすく使いやすいから
- 慣れだけではないと思うけど、やっぱり Fortran に戻ってしまう。
  分野では未だに主流でもありますし。
- 計算速度が速く、これまでのプログラム資産が多い
  文字列は苦手なので、perl などの他の言語と組み合わせて使うことが多い
- Fortran で書かれたコードがたくさんある。言語仕様の違いから,C/C++と比較して手動で最適化する必要が少ない。
  - 慣れているから.

- 過去に作られていたプログラムを引き継いでいるため.
- 過去のプログラムが豊富にある
  視覚的で間違いにくく、プログラムしやすい
- 大学の講義である課題演習で使ったものが Fortran で、そのまま使っています。
- C 言語などに比べて比較的容易に理解できて便利
- 最初に学んで、研究をするのに十分なため。また、他の言語よりも計算が高速と言うことなので。使う数値モデルが fortran なので、ソースの改変を行うため。
- 開発コードが Fortran のため。
- 気象業界では、Fortran 言語が最も使用されているため。
- 必要だから使っている
- プログラム資産が fortran だから、使い続ける.
- 職場で伝承されているサブルーチンを使うため。
- 大学の研究(気象学、気候学)で最初に習ったのが Fortran だった。その後、特に他の言語を使用しなければできないようなことをやっていなため、Fortran をそのまま使い続けている。
  大学の研究室(気象学・気候学)では、複雑な統計解析を行うプログラムを fortran のコードで作成し、それは研究室全体の財産として後輩に受け継がれる。もちろんそれは分野としての財産でもある。その意味でも Fortran は重要。
- 関係機関から頂く参考プログラムのソースや、社内の過去の資産プログラムが Fortran ベースであることが多いため。
- 既に大規模な計算コードが整備されている。スーパーコンピュータにおける性能が高い。また計算機ベンダー側もチューニングしている。
- 自分では並列化プログラムは書きませんが、並列化されたプログラムを使ったシミュレーションを実行することがあります。
- 配列演算が Fortran (90 以降) ほど綺麗に書け、かつ、コンパイラが最適化してくれる言語はない。優秀なコンパイラがいくつもある。
- 慣れているので一番使いやすい。数値計算に特化しているのも好きです。
- 「慣れている」から、というのが本音だが、研究分野では標準的であることと、可読性が高いと思うことも、理由の一つ
- 従来からの豊富なプログラム資産が利用できる。
- 自分の身近のプログラムコードの大半が Fortran だから
  - 単純に使いやすい/わかりやすいと思う
- 身体に染みついている。日本語を日常言語として用いるのと同じ。

- 数値計算に非常に適しているから。
  実行速度が速いから。
- 配列計算が楽
- C 等と比べて簡単。
- 数値モデルの多くが Fortran で書かれており、自分自身が最初にマスターしたのも Fortran なので、なじみやすいから
- 気象分野の計算では依然として主流であるから。他の言語を使う予定は基本的にない。
- 言語仕様がシンプルだから
- 特定のプロセッサ上ではもっとも手軽に効率の良い出力が得られるという点と,過去の資産(おおむね Fortran77-like な 90 のコード)の利用.
- Simple かつコンパイラ性能を生かせる
- ソフトの資産が残っているから。また、別の言語に書き換える理由がない。さらに、コンパイラの最適化がされやすく高速で動くから。
- 人間の思考に近い気がするから使っています。
  (例: 配列が 1 から始まる。c などでは 0 から)
- 長年使い慣れた数値計算言語であるから。
- 気象を専門としているが、気象学では Fortran を使うことが基本である。
- 自身の研究でシミュレーションがしばしばとりあげられるため。
- 使う理由
  仕様が単純で、大型計算機でのコンパイラも充実しているため。
- 使う理由:過去のコードを改善改訂しつつ利用。C 等の言語に置換するにはステップ数が膨大でできない。
- 理由 1:大規模シミュレーションを実施する際に性能が出やすいと聞いているため。
  理由 2:共同研究者らが開発したコードを拡張して自らの研究に使うコードを作成しているのだが、現状では、そのベースとなるコードが全て fortran であるため。
- 過去の遺産があるから
- 多次元配列の形状を動的に外部ルーチンに渡す機能が必要なので(Fortran77 でサポートされている整合配列など)、
- 科学計算に強い。同分野の研究者の多くが利用しているため、情報交換の際にも効率がいい
  - 最初に習った言語
  - 研究をはじめた時、周囲では Fortran が使用されていた

・　他所から借用したり導入したりするプログラムが主に Fortran で書かれている

- これから Fortran 以外のものを習得して得られるメリットと、それに費やされる時間と労力のデメリットを比較して、現状まだデメリットの方が大きいと感じるため。
- 性能。過去の資産流用。
- 既存のサブルーチンの資産が活かせるから
- Fortran 以外に使いやすそうなシミュレーション用の言語を知らないから。
- 使う理由:業界におけるプログラム群が、基本的にフォートランで書かれているから(海洋・気象系)。また、初めて学んだプログラム言語がフォートランだから。
- 研究で必要だから。
- スパコンでの利用に適していると思うから。
- 人事異動でプログラマーじゃなくなったからです
- とりあえず、計算機能が安定していて使いやすい。他の言語を覚える必要性が今のところない。
- 過去の膨大な試算があり、他の言語に変換する原資が無い。
- コーディングが簡単
- 既存ライブラリー資産の活用で使っている。
  オブジェクト指向を原点とする言語ではないので、他の言語を使っている。コンパイラーのベンダーサポート(セミナー等を含む)が少なく、言語に発展が見られないので、他の言語を使っている。複素数計算は、Fortran を選ぶ。
- プログラムが見やすいので。
- 使う理由
  コンパイラの最適化機能が最も高い
  過去の資産を活用できる
  複素数、多倍長が気軽に使える
  使わない理由
  アセンブラやコンパイラ組み込み関数等を活用した先進的な最適化を行う場合にはＣを使う必要がある。
- Fortran しか知らないため
- 使う理由:計算が速い
  他の言語も使っているので、特に問題ない
- 使う理由:　数値計算の業界では Fortran をお使いである人が多く、共同して研究を進めるときには言語を統一しておいた方が進めやすいと考えられるため。逆に言うと、周りがC/C++であればそちらを使います。

- 過去の遺産が多くあり、主要なコードが Fortran である。
- 過去資産の利用のため
- 私は流体系のシミュレーション(主にスーパーコンピュータ)とデータ可視化(PC とワークステーション)で計算機を使います。シミュレーションプログラムでは Fortran を、可視化プログラムでは C++を使います。どちらもプログラムの書きやすさとメンテナンスのしやすさの点で最善の選択だと思っています。(他にも複数の言語を一応は知っています。)全てとは言いませんが、多くの計算機シミュレーションでは、プログラムを書く言語としては今後も Fortran(90 以上)が一番よいのではないかと思います。これはたとえライブラリなど過去の遺産を全く使わないとしても、です。つまり DSL(Domain Specific Language)として Fortran は今後も重要となると思います。Fortran の利点(カプセル化(Private/Public)など大規模なシミュレーションコードとして必要な機能を備えながら、余分な機能がないことです。ある機能を実現するのに複数の実装方法がありうる C++よりも、あまり選択の余地のない Fortran の方がシミュレーションプログラム言語としてはよいと思います。この意味で、C++は(高機能ではあるものの)複雑過ぎます。
- 過去の資産があるため

Q9: Fortran 言語とそのコンパイラ、並列化プリプロセッサーなどについて、改善して欲しい点や要望等をお聞かせください。

- 分散並列は MPI が業界標準になってしまったので、その代替を考えるよりも、スレッド並列をメニーコア向けに高効率で自動並列化するコンパイラ・プリプロセッサの開発に注力して欲しい。
- バグが起こった場合の原因の同定や、通信・キャッシュチューニングのための情報取得などをもっと容易にするコンパイラ・性能評価ツールの整備。
  コンパイル時に不確定な入力パラメータをループ長とするループに対する最適化を適切に行うために、コンパイル時にパラメータを仮に設定できるような仕組みを fortran 標準として用意すること。
- Fortran が絶滅する(というか、大規模計算機に向かなくなる)という状況になった場合は、代替言語を普通に使うことになると思います。
  Fortran も仕様の中に最新の思想がどんどん入るべきですが、コンパイラがプリプロセッサ的に翻訳する程度では、新しい思想を入れた意味は半減してしまい、積極的に利用しようというモチベーションはどうしても低下してしまいます。
  並列化プリプロセッサも同様です。
  コンパイラによる最適化が進まないと、言語仕様をいくら拡張してもユーザは増えないと思います。
  勿論、開発効率が圧倒的に改善する場合は別ですが。。。
- 詳しいことは知らないので回答できません
- なんでもできる言語でなくてよい。
  大規模数値計算に関しては、他の追随を許さない、最速の言語となることを期待しています。
- 自動並列化の性能向上
- もっと容易に並列化ができるとありがたい
  - 自動並列化はされたとしてどれくらい速くなりそうか(全く速くならない場合もあるため)、どこをどうすればもっと速くなるかなどが、コンパイラ時のメッセージである程度分かると試行錯誤に掛ける時間が少なくなる。
  - HPF は一体どうなったのだろう?
  - MPI 並列化でデッドロックに陥った際の適切な解決法(問題の箇所や問題の性質など)を提示して貰えればありがたい。
- 要望とは異なりますが、ユーザー側の工夫として。

計算機環境が変わるとコンパイルが通らなくなったり、実行時にエラーが出たりすることがあるのですが、どうも古いコードである場合が多く、プログラムの書き方に問題があるのがほとんどです。標準的なコーディングルールを決める等、ユーザー側でも絶滅させない工夫が必要かと思います。これからプログラミングを始める学生にとっても、どう書いたら分からない状況からスタートするよりは、コーディングルールのような手引きにできる資料があるだけでもだいぶ敷居が下がるのではないでしょうか。

- 特になし
- ベンダーによるコンパイラオプションの違い(「方言」)をできるだけ減らしてほしい。
- C++で書いても、Fortran で書いても、お互いに翻訳するソフトを開発してほしいです。
- 配列受け渡し・参照の単純化
- ベンダー間でコンパイラーのきつさのレベルが違うのを何とかしてほしい。
- 多様なファイル入出力にも対応できるようにして欲しい。
- Linux Fortran (gfortran)を使う機会があるが、標準的でないのでよくわからないエラーが出る。Linux は汎用なので、どこかで改良ができないか。
- 特にありません。
- とにかく速くしてくれ
- Fortran ばかりでなく C/C++の高速化に力を入れてほしい。
- GPU 上でもコンパイル可能になると良いと思われます
- Fortran を他言語に比べて劣るかの如き、或いは絶滅危惧であるかの如きに言うのは控えて貰いたい。
- 自動並列機能の強化。
- もう少し簡便に開発ができる並列化言語が必要だと思う。
- 特になし。
- CoArray がよさそう、という予備文献調査での個人的印象をもっているが、講習会などの訓練と教育の機会が少ない事が残念。
  GPGPU を直接たたける Fortran があればうれしい。
- 使わない
- フリーのコンパイラがデフォルトで PC に入っていたりするとありがたいと思う.
- 処理系による微妙な仕様の差異、マニュアルを見ないと意味不明なエラーメッセージ、コンパイラ自体のバグなどで生産性が下がっています。
- Portability and 'easy transfer (transplant)' are always a problem in any language. Every time I get a faster machine, my work slows down for adapting the existing FORTRAN program to a new dialect environment..

- 特になし
- GPU 対応の自動並列化がまだフリーソフトウェアではできない。
- 特になし
- ある研究所のスパコンでモデル計算を日立の FORTRAN で行ったことがあるが、コンパイラが優秀でプログラムを書くときにあまり工夫しなくてもパフォーマンスが高い実行ができた。こういうコンパイラは今後も必要だと思う。
- さまざまなアーキテクチャを使う現在では、define 的にいろいろ宣言したい(一部可能だが)。
- 特になし
- ??
- Fortran の方言問題の改善
- 個人的に書き留めたのがありますので、コピペで失礼します。最新版を追いかけていませんので、ひょっとすると実現しているのかもしれませんが。
  サブルーチン・関数の中のサブルーチン・関数を許す(変数のスコープの自由度)
  内部関数を関数引数で渡せない
  変数宣言文を途中で書くことを許す
  配列演算の順序の明示法の実装
  デフォルトの配列範囲の指定をデータを読み込んでから設定する
  implicit none のデフォルト化
  コンパイルオプションをソースに書く方法。
  関数の充実(効率 2 の次でいいけど、高効率の関数に簡単に置き換えられる=いわゆる遺産をどんどん取込む)
  整数型で割る計算式に Warning を出す。
  コンパイラメッセージの日本語化:日本語 Fortran
  return に返す数を書く
- 特になし
- 古いバージョンしか知らないので、今では改善されているかもしれませんが、継続行の指定の仕方、命令文の後の同一行に継続を指定できるようにする。
  命令文の同じ行の後にコメント文が入れられるようにする。if 分の書き方、=、<、>、>=、等を使えるように。1 行の字数を 200 ぐらいに(もうなっている?)。
  COMMON 文の使用を止める(後の誤りの解読が大変)。配列宣言を、配列が出てくる前の任意の場所で可能に。
  IMPLICIT 文による変数の型宣言を止める。

使用するすべての変数を型宣言する(あるいは変数名の打ち間違いの発見法の導入)。
簡単なグラフ表示の充実。横軸(1 変数)と縦軸(複数変数)を与えて、点、あるいは折れ線のグラフ、できれば 2 次元変数の等高線表示ぐらいで良いので、SSL に入れる。

- (1) C++にある、インライン関数のようなものがあると嬉しい。現在は、パフォーマンスを挙げるために、プリプロセッシングを使っていますが、型チェックなどの各種チェックが入らないので、バグの混入可能性が増える。
  (2) あまり、パフォーマンスを落とさずに、様々な subroutine をパラメータファイルでコントロールするために、C++などにあるように、関数をポインタを通して呼び出せるようにして欲しい。
- 77 と 90 の保存
  プロットの為の世界的なライブラリーの構築
- 特に無し
- Fortran 開発者(ベンダー)が付加価値を付けるという意味で欲しい点や情報:
  - 各科技計算に基本的な処理の関数化と最適化
  - コンパイル最適化レベルと計算結果の関係性
  - 数値計算の一般論でなくコンパイラレベルでの理屈付け
- 機能は別にそのままで良いが、C と同程度の高速性を保ち続けて欲しい。
- 特に要望はありませんが、絶滅しないでほしい。
- Fortran からはデバイスが見えない。アセンブラやキャッシュまで想定したコードを書こうとしても、それがかなわない。変数のスコープが小さくできないためだ。
- Coarray の良い教科書がない。
- format の簡略化
- module 化してからついていけなくなってきています。もう一度、ちゃんと勉強しなおさなくてはならないと痛切に感じております。その時に Fortran である必要があるのかは、悩む所です。
  プリプロセッサは以前は C を通さなくてはならなかったかと思うのですが、今の版は簡単になっているのでその点は便利で良いです。
- 特にない。
- 言語としての抽象度・自由度を高めることよりも、高い計算実効性能が達成できそうな範囲の記述力に特化することで、他言語との差別化を明確にして欲しい。(同じアルゴリズムでも、一般的な C コードではコンパイラの最適化が難しいけれど、Fortran であれば容易に高性能が実現される、のような。)
- 特にない。

- 数学ライブラリがあれば現状で満足しています
- 特になし
- Fortran 77 を一貫して使っている。最新の Fortran に比べて制限が多く柔軟性は欠けるが、そのように固定化されることで、かえって使いやすく、ヴァージョンなどによる違いを意識せずに、プログラミングに専念できる。人によってニーズは異なると思うが、むやみなアップデートはむしろ有害ではないかと思う。
- 宣言文などが略語で短くなってくれると有難い。
- プリプロセッサの仕様が完全に C 言語からの借り物なので、あまりイケていない気がします(改行の扱いなど)。
- まれに最適化の段階を変えた時に結果が変わることがあるが,それをより起こらなくしてほしい.
- Testing Framework
  Parallel Debugger
- 今のところ、思いつかない。
- 様々な OS で同じように使えると非常に便利。
- 文法的な融通の利かなさ、柔軟性のなさを解消してほしい。
- 商用コンパイラのライセンス料金の値下げとポストプロセス用の可視化ツールの充実。
- Cuda fortran が pgi 社のみでしか提供されていない点
- 特にない
- 古い言語なので、記述法がいくつも存在するのは煩わしい。学生に教えにくいです。
- FORTRAN の解説書、特に 2003 以降の規格に対応した書籍がほとんど皆無。
  RAD 開発環境の充実
- 現状で十分です。あとは一般にわかり易くする「説明」の提示の仕方を考えて欲しいです。他方、「一般に普及するために、コマンドや処理方法をかえる」ということは絶対にしないで欲しいです(あくまで今の状態で動かせることが条件)。
- 絶滅させないでほしい
- JIS7000->77->90/95 と大きな変更があって着実に使いやすくなっているが、非並列化部分ではもう大幅な変更は不要である。
- Fortran77 が使える環境を、将来にわたって残して欲しい。
- Fortran2003 のフルサポート
  unsigned int のサポート
  リスト、スタック、連想配列のサポートとポインタの廃止

- 引き続き,高速な実行モジュールがコンパイルされるよう最適化技術を改善していって欲しい.
- 並列計算等についてわかりやすい最新版の書籍が欲しい。
- グラフィック可視化の簡単化
- 少なくとも現状の環境が維持されることを望みます。
- Coarray をまともにサポートしたコンパイラのリリース
- windows のように、気軽に使えるソフトウェアができると、他のユーザーへ拡張が期待できると思います。
- 新規機能の取り込みよりも,バグのない安定したコンパイラを継続的に提供して欲しい(できるだけフリーで).
- その域まで使いこなせていないと思いますが、より単純に高速化、大規模化ができるとよいです。
- 静的型付けによる安心感や、最適化による処理速度の速さが Fortran のメリットであると思われるので、この点をないがしろにしなければ使われ続けると思います。しかしながら、プログラムを記述するにあたって、新旧混ざっていくつもの書き方が存在し、混乱します。複雑な記述で、新しい言語にあるような機能を追加するよりも、単純で統一された記述で高速処理、並列処理ができることを望みます。
- 現在オブジェクト指向プログラミング言語の世界で蓄積されている、計算手法、性能向上のためのノウハウを活用すべきであり、Fortran に閉じた孤立した世界の技術は存在価値が低いと思います。
- とくになし
- 特にありません.
- Visual Fortran よもう一度。要するにインストールが容易でデバッグツール、ライブラリと一体化したコンパイラ環境がほしい。
- 1 行最大 132 文字の拡大
- 最低限の文字処理の標準化。現在の標準化には問題があると聞いているので。
- コンパイラの FORTRAN77 への対応は今後も継続していただきたい
- もう 2015 年なのに、fortran2003 にすらまっとうに解釈できるコンパイラが少ない時点で、やる気が感じられません。
  粛々と滅ぶ運命にあるのでしょう。その原因は結局ベンダにあります。
- fortan77+95 くらいで満足して使ってます。
- Cuda 対応のコンパイラが無償だと助かります

- 並列化計算の方法・コードの書き方・準備に関する初心者向けの入門書・情報をさらに充実させてもらえればとてもありがたいです。
- 絶滅させないでほしい。
- 通信を含む速度計測ツール(遅い個所・そいの理由を示してくれるツール)を改良してほしい。キャッシュミスやレイテンシ等によるものなのか、何の最適化の問題なのか、等。たとえば、キャッシュミス(そのレイテンシ)がないとしたときの速度を推定するツール(シミュレータ)、MPI で遅いプロセスがなぜか待たされるような場合の理由を検出する(状態の表示をする)ためのツール、など。
  スパコンの BMT などでベンダがシミュレータを使っていると思いますが、各コンパイラについて、どのマシンでも、ユーザに容易に使えるようにしてもらえるとうれしいです。
- 容易に導入できる Fortran 用の統合開発環境が欲しい。Eclipse はあるが敷居が高くまだ試してない。
- コンパイラ自体には特に要望はないが、2003、2008 の良質なドキュメントが欲しい。
- 構造体データが C 言語並みにハンドリングしやすくなればありがたい。
- 暗黙の型宣言はなくなったようだけど、暗黙の装置番号 5 番、6 番は今後もなくならないのだろうか。使い慣れたけど、その点の自由度の無さだけは Fortran が悪く言われるひとつでもあり、変えない理由はあるのだろうかと思う。
- あるコンパイラに気に入られるコードにしようとすると指示行をたくさん入れないといけなくなり、コードの可読性が下がってしまうのをなんとかしてほしい(なんとかしたい)。
- 開発環境が悪い(Visual Studio のような開発環境が使えない).
  フリーのコンパイラが少ない.
  文字列操作が不便.
  C 言語のような例外処理.
- 旧態依然の暗黙のルールがあるため、それらを取っ払って頂きたい。仮に 90,95 以上のアップデート版にカバーされていれば無視してください。
  世相とのズレにより不要、というのは愚直だと思われる。
- 識者の方に Fortran ではデバッグが Matlab に比べるとずっと大変なので,標準的な手法やデバッグに強い書き方をまとめていただけると大変ありがたいです.現状は On the job training で覚えるしかなく,並のレベルの学生では自得するのはなかなか難しいです.
  - より使いやすいビット演算
  - 文字操作
- 分かりやすいバージョン別の機能マニュアル
- 自動的に並列化した上に性能もでるようにして欲しい

- 特になし
- 便利な組み込み関数を使うと、逆に効率が落ちる、ということがないようにしてほしい。
  また、コンパクトなソースでも最適化が可能なコンパイラにしてもらいたい(ユーザーが、最適化のために、わざわざコードを冗長な記述に書き換える必要がないようにしてもらいたい)。
- C の C++ のように,ラッパーや使用できるライブラリが増えると良いなと思います.そのためには,ユーザー数が増える必要があると思います(メンテナンスにも関わる).ユーザー数を増やすためには,使いやすさ・わかりやすさが優先事項であると考えられます.使いやすさ・わかりやすさは,良い解説書の有無に関係があると思います.良いドキュメントが豊富であれば利用者は独習しやすくなり,結果としてユーザー数の増加につながるように思います.
- 綺麗に書いて欲しい
- フリーのコンパイラの一層の充実と、マトリックス計算の Matlab のような計算の簡素化。
- CUDA Fortran の PGI の独占化がなくなってもう少しオープンになってもよいのではないかと思います。co-array のような先進的実験機能は Fortran において大いに推進してもらう価値があるとおもいます。
- 初心者向けにわかりやすい解説が欲しい。
- 計算速度の向上以外には特に要望はありません。
- バイナリファイルの取り扱いが不便。
- Fortran 2003 以降が浸透するのか危惧している。
  自分の周りでは 2003 以降の文法を使う人は周りにほとんどいない。
  これはベンダのコンパイラが対応していないせいもあるかもしれないが、和書のテキストが存在していないなど、情報の流通量が少ないせいも考えられる。
  初学者にとっても、日本語のテキストがない言語を学ぶのは敷居が高いだろう。
  web 上で日本語のテキストを無料配布するなどしないと、今後の利用者数の維持は難しいだろう。
- 特になし
- 名前空間や,サブルーチンの名前の規約(内部的表現)などをコンパイラ依存にしない.混合言語プログラミングがしやすいように.
  Interface 文はもう少し簡便にかけるようになっているともっと良い.
- 構造体の配列のメモリ配置を選択できるようにしてほしい。
  要素配列を連続にする場合と構造体の要素を連続にする場合など。
- GPU 対応の汎用性のあるコンパイラ

- できれば GPU を意識せずにプログラミングできるような GPU 対応のコンパイラ
- Linux 以外、例えば FreeBSD 等でも商品のコンパイラや環境が利用できれば。
  お試し版の使用制限も減らして、裾野が広がるように努力して欲しい。
- 特にはありません。C 言語の null のような文字列の終端符号があれば、もっとプログラムが便利になる。
- 商用コンパイラには,Fortran 2008 の仕様を確実に実装する。clang に相当する LLVM ベースのフリーのコンパイラを開発する。
- Fortran をずっと提供しつづけてほしい
- 並列化プリプロセッサなどについて学べる初歩的な教科書が少ないように感じます。初学者向けの資料を整えて欲しいです。
- 高速化。
- 他言語、描画ソフトとのインターフェースの整備
- 文字処理が面倒でしたが、最近は改善しているのでは。
- ボランティアによる委員会で言語仕様が決まり、そこにコンパイラメーカーの人達の発言が強すぎるため、明らかに整合的で望ましい機能でも、「厳密な規格を書くのが大変」「実装が大変」などの理由で、先延ばしにされたり廃案になったりする。(例えば、「定数」の右辺に使える定数関数の範囲がなかなか広がらない。)
  改善の要望: 関数型プログラミング機能の充実(オブジェクト指向なんかより、ずっと数値計算に有効です)、右辺から型を推定し宣言を省けるようにする(Haskell や最近の C++ を参照)、parameterized type の充実、局所変数の定義(例えば、do ループの中だけで有効な変数や定数を宣言できるようにする。C++参照)、optional 属性を OPEN などに伝播するようにする、可変長文字列を完全に言語に組み込む、OS の機能を何らかの標準の下に Fortran から使えるようにする(ファイルの rename、cd, ファイルの削除、など。POSIX を採用するとか)、配列以外の container の標準化(いまどき dictionary や伸び縮みする配列を自分で書かなくてはならないなんて . . .)、lazy evaluation(あるいはそれと似た機能)の導入(C などの short-circuited AND や OR, 配列を返す関数の連鎖のために大きな一時記憶が必要になるのを避けるため、など)。
  要は、21 世紀に相応しい言語になって欲しいということです。また、Fortran は「計算機でもできる」ことを人間にやらせる程度が高すぎます。(例えば、character(*), parameter:: a(:) = ["a", "bb", "ccc", "hello"] と書けば character(5), dimension(4) の配列だと決定してくれればよい。) それから、Fortran で書くと余りに面倒なので、ほかの言語で書いて、その中から Fortran を呼び出すことになりがちなのを全部 Fortran で書けるようにして欲しい、というのもあります。

- サーバー用のコンパイラはあるけれど一般の PC に入れるコンパイラはないという場合があったりしますが、サーバーでも PC でも同じ会社のものを使えると便利なので、個人 PC 用のものも(需要は少なそうだけれど)積極的に作ってもらえると助かります。
- コンパイラの「方言」は極力無くして欲しい
- Fortran コードの他言語(C や VB など)への翻訳機能
- 入出力データフォーマットに関して、もう少し自由度があった方が使いやすい。
- 特になし
- Fortran の強みはスーパーコン上の(ベンダの)コンパイラの最適化にあると考えるが,実態としては Fortran77,90-like なコードが速いだけであって 95 以降ましてや 2003 の OOP スタイルは遅い.新規の開発でも古い資産の利用がまず考えられるため,このような状況を招いているとも言えるが,コモディティ化するプロセッサ(CPU,GPU,etc.)上では現代的な言語で現代的な記述をすることが最適化に繋がることを考えると不毛である.そもそも Fortran2003 以降の言語仕様はお世事にも良いとは言い難い.
- 絶滅しないで!
- 科学技術を発展させるために京コンピューターがあります。しかし、それを使うには並列アプリが必要です。そのアプリを作成、修正するためには Fortran が必要です。その Fortran を教える機関がいまやなくなっています。科学立国日本を継続するためには Fortran を復活させる必要があります。大学機関で教育をしてもらいたいです。
- 今後も Fortran77 が利用可能な状態を維持して欲しいです。
- 特にございませんが、気象の分野で Fortran がなくなることはありえないと思います。
- とくになし
- 特になし
- コンパイラディレクティブを使った並列化手法は継続すべき、分散メモリ型の並列化手法が HPF/JA 並みのものがほしい。
- 文字列処理の改善
  - Fortran の言語仕様追加はもう必要ない。むしろ追加を止めて欲しい。
- 他言語へのコンバータを開発していただきたい
- 行列計算など簡単になったが、エラーが見つけにくくなった面もある。コンパイラーでチェックできそうな気がするときもあるので、注意深くチェックしてほしい。
- (どこにも書く欄がなかったので)特に Q3 の設問がおかしい。多くの技能は、一つの方法で取得できるとは限らない。また、「マスター」という、人によって大きく定義が揺らぐ言葉を設問に用いていることも大きな問題である。全体的に、複数回答したいができない項

目が多く、非常に答え難かった。このようなアンケートを躊躇なく出してしまう神経をまず改善して欲しい。

- 特になし。
- 初心者でも使いやすい並列化プリプロセッサーがあると、自分も、学生教育の観点からも、たいへん助かります。
- HW ベンダー以外が作る並列プリプロセッサは、機種移行の助けになるのであれば価値があるので、主要 3 アーキテクチャ(x86、Power,SPARC)くらいはサポートしてもらいたいところです。

  ベンダーのコンパイラは、正直変な機能よりちゃんと動いてくれたほうがいいです。
- バージョンが新しくなると複雑になってきているような気がする。
- Oracle が Java でやっているような、MS が C#でやっているような、Apple が Swift で始めたような、コミュニティー主導で、プログラマーが書かなくても良い API を増やして欲しい。
- せめてコンパイラ組み込み関数は Fortran でも使えるようにしてほしい。
- 特にありません
- 特になし
- 特にありません。
  - 言語の名前が悪いと思います。プログラマの多くは現在の Fortran を知らないので、おそらくは古い FORTRAN(77 あるいはそれ以前)を思い浮かべて内心馬鹿にするという傾向があると思います。このような些細なことが言語の普及拡大を制限する面があると思います。Fortran90 の時に別の言語名にすればよかった。たとえば(もう使われてしまっていますが) "F" とか。
  - Fortran2003 以降、言語仕様が複雑になりすぎる傾向がよくないと感じます。あらゆることを Fortran でする必要はありません。Fortran 言語仕様の複雑化は仕方がないのかもしれませんが、その場合でもスーパーコンピュータ向けには、DSL としての機能を限定したサブセットを提供していただくと助かります。(この意味で "F" のアプローチは魅力的に思えます。)