2021 年度若手研究者アンサンブルグラント
新規課題（第１ステージ）の公募について

東北大学研究所長会議 代表
加齢医学研究所 所長　川島 隆太

研究所若手アンサンブルプロジェクトWG リーダー
材料科学高等研究所　甲斐 洋行

東北大学附置研究所若手アンサンブルプロジェクトでは，学内の若手研究者による連携を促進するために，複数部局の研究者で構成された共同研究グループへ研究費を配分いたします。本研究費は，全研究領域を公募対象とし，個人の自由な発想に基づく部局間連携によって生み出される萌芽的な学術研究課題に対して助成を行うものです。新たな研究のスタートアップ，あるいはこれまでのテーマの幅を広げる新展開への試行を奨励する課題を公募します。若手研究者による応募を歓迎しますが，共同研究者として研究グループへ参画する方については，要項に記載された所属の要件を満たしていれば，身分等は問いません。新しい着想や視点（研究内容はもちろん，他部局設備の利用による研究の効率化なども対象となりえます）を基に，積極的な応募をお待ちしております。

公募要項

【研究期間】
2021 年 6 月 1 日（予定）から 2022 年 3 月 31 日。

【支援内容】
1 課題最大 50 万円，15 課題程度。採択後一定の期間を経て，研究代表者の所属する東北大学附置研究所・センター連携体の各部局に対し，本年度の世話部局である加齢医学研究所から配分されます。

【対象となる課題】
本学の複数部局（研究所，センター，研究科等）に所属する教員・研究員で構成される研究グループによる研究課題。全領域の研究を対象とします。異分野融合研究，学際研究が必須条件ではありません。

【対象となる申請者】
申請者（研究代表者）の対象は，東北大学附置研究所・センター連携体の各部局に所属するポスドク，助手，助教，講師，准教授（特任・特定を含む）とします。特に若手研究者の応募を歓迎します。研究代表者以外の共同研究者については，職名・身分の制限はありません（学外の研究者も可とします）が，学生の卒業・修了などにより，複数部局のグループが研究期間の大半に構成されなくなる見込みが明確な場合は，対象とはなりません（後期課程などへ進学希望，ポスドクとして在籍予定などの場合は対象とします)。

- ここで「東北大学附置研究所・センター連携体の各部局」とは，金属材料研究所，加齢医学研究所，流体科学研究所，電気通信研究所，多元物質科学研究所，災害科学国際研究所，東北アジア研究センター，学際科学フロンティア研究所，材料科学高等研究所（AIMR)，電子光理学研究センター，未来科学技術共同研究センター（NICHe)，国際放射光イノベーション・スマート研究センターを指します（以下，同じ)。
- 本公募では，兼任・兼担などの場合 (学際研のメンター制も含む)，在籍するあるいは主な活動拠点である東北大学附置研究所・センター連携体の各部局に所属する研究者とグループを構成しても，それ自体では複数部局とはみなされません。
- 申請者（研究代表者）は，上記の各部局のいずれかにおいて本学の予算管理システムを使用可能であることが必須です。
- 応募は 1 人 1 件のみ（研究代表者・共同研究者あわせて）とします。
- 申請代表者・分担者のメンバー構成が申請対象に該当するかどうか判断が難しい場合は，締切前に余裕を持って若手アンサンブルプロジェクトワーキンググループ（WG）にご確認ください。

【選考】

萌芽的な研究を発掘し多様な研究を支援するために，研究内容についてスクリーニングを WG で行ったうえで，15 件程度をランダムに採択し，研究所長会議で決定のうえ，2021 年 6 月下旬頃に選考結果を通知します。

申請内容のスクリーニングでは，下記のいずれかに該当する申請は採択の対象外となります。

- 募集要項を満たしていない申請
  - 例： メンバー構成が複数部局に該当しない場合。申請者自身で明確に判断できないときは，**締切前に余裕を持って** WG に確認してください。
  - 例： 申請代表者が東北大学附置研究所・センター連携体の各部局に所属するポスドク，助手，助教，講師，准教授（特任・特定を含む）でない場合。
  - 例： 申請書が 3 ページ以上の場合。
- 以前に採択された自身の研究と同一または酷似する内容の申請
  - 以前に採択された自身の研究と類似していると判断されうる研究課題の申請については，以前の課題との違いを申請書の「過去の採択課題との相違点」欄に記入してください。
- 最低限の研究内容が示されていない申請
- 必要経費内訳に正当性の無い申請

【来年度の研究継続】

(i)本年度の採択課題のうち希望するグループ，および (ii)新たに申請された研究課題を対象に，2022 年 1 月頃に開催予定のシンポジウムにおいて参加者全員と世話教員によるピアレビューを行い，来年度の継続課題（研究期間：2022 年 4 月～2023 年 3 月，研究費上限 100 万円）として 2～3 件程度を採択し，2022 年 4 月に決定する予定です（図 1）。また来年度もこれらの審査採択課題に加えて，15 件程度の新規課題（研究期間予定：2022 年 6 月～2023 年 3 月，研究費上限 50 万円）を採択する予定です。なお，同一課題での継続は 1 年度まで（新規採択 1 年度＋継続 1 年度）とします。

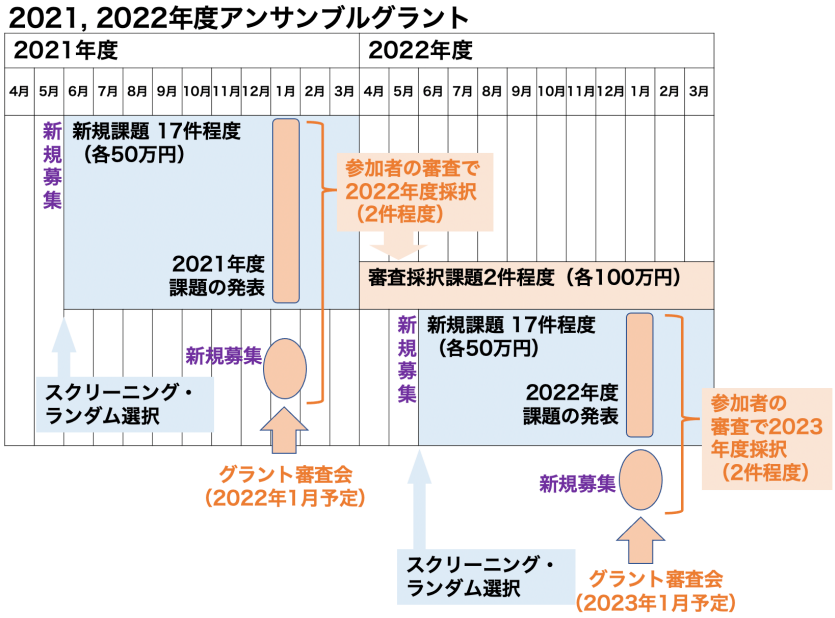

図1　採択プロセスと研究期間

【応募方法】
所定の書式を用いて申請書を作成し，PDF に変換のうえ，締切日までに研究代表者が下記 URL のフォームより送信してください。

https://forms.gle/5Q5M8axSALHGEvqR8

締切日：2021 年 6 月 4 日（金）
異なる書式によるもの，あるいは提出期限をすぎたものは受理されません。

【報告】
研究期間終了後，所定様式の成果報告書の提出が義務づけられます（成果報告書の内容は若手研究者アンサンブルプロジェクトのウェブサイトで公開されます)。また，本年度中に開催予定のワークショップで研究の着想と成果を発表していただきます。なお，成果の公表の際には，本公募プログラムの支援によるものであることを記載してください。

【取扱い】
安全衛生管理ならびにネットワーク管理，研究不正防止，法令順守などについて，本学ならびに所属部局にて実施運用しているすべての規則・指導に準拠して研究を実施していただきます。なお，これらを逸脱していると判断される場合には支援を中止させていただきます。

【連絡先】
本公募に関してご不明な点は，

- 東北大学研究所若手アンサンブルプロジェクト ワーキンググループ
  ensemble_secretariat@fris.tohoku.ac.jp
- 材料科学高等研究所 甲斐洋行助教
  kai@tohoku.ac.jp

までご照会ください。

申請書の書き方について

申請書は適宜字数を調節して１～2 ページに収めてください。

１．研究組織

- 本学の複数部局（研究所，センター，研究科等）に所属する教員・研究員・技術職員で構成される研究グループとしてください。これ以外に，分担者であれば学外者を含んでも結構です。研究代表者名の前に，◎を付加してください。
- 所属部局と主な活動部局が異なる場合（複数部局の兼任の場合など）は，所属部局と主な活動部局の両方を記載してください。

２．研究経費

- 研究経費は設備費，消耗品費，旅費，謝金・人件費で本研究の遂行に必要なものに限ります。研究室運営のための経費や，他の研究の経費として計上することがふさわしいと考えられる支出は認められません。

３．研究内容

- 以前に若手研究者アンサンブルグラントに採択された自身の研究と類似していると判断されうる研究課題の申請については，以前の課題との違いを申請書の「過去の採択課題との相違点」欄に明確に記入してください。
- （継続を前提とせず）1 年分を記載して下さい。

４．他の研究費申請について

- 本グラントは他の研究費との重複申請を制限しませんが，他研究費に制限がある場合には，考慮のうえ申請してください。

応募課題のランダム選択の手順

1　申請書を受理した順番で，1 から始まり 1 ずつ増加するエントリー番号 1, 2, 3, …, N をすべての申請書に付与する。

- 期間内に再送信した場合や，提出後取り下げた場合についても，最初に申請書を提出したタイミングでエントリー番号を付与する。
- エントリー番号は，受理あるいは募集締切りの時点で申請者に通知される。

2　募集要項に基づいて，申請書のスクリーニングを WG により行う。採択予定件数 M（$M = 15$ 程度）を決定する。スクリーニングを通過した課題数が 15 件を大きく超えない場合は，金額を調整のうえ全件を採択する場合がある。スクリーニングおよび採択予定件数の決定は，2021 年 6 月 10 日（木）までに行う。

3　スクリーニングを通過した申請書のエントリー番号について，添付の Python スクリプト（Python 3.7）を使用してランダムに順位付けをして，上位の M 件を採択する。

3.1　ビットコインブロックチェーンにおける，**2021 年 6 月 11 日（金）午前 8 時 0 分（日本時間）以降で一番早い順に 5 個のブロック**のブロックハッシュの和を乱数シード S とする。`random.seed(S)` により乱数を初期化する。

3.2　エントリー番号を昇順に並べたリスト ENTRIES を用意する。

3.3　`number_order = random.sample(ENTRIES, len(ENTRIES))` によって発表番号 ENTRIES の順番をシャッフルする。

3.4　number_order の順に，M 件を採択する。

4　採択課題の決定・通知時に，2 のスクリーニングを通過したエントリー番号の一覧と，3.1 で使用した乱数シードは公開される。

【補足】

1　ビットコインのブロックは約 10 分おきに新しく生成される。ブロック生成のたびに block height は 1 ずつ増加し，ブロック固有のハッシュ値（32 バイトの数値）が決まる。ブロックハッシュ値は以下のような性質を有するため，ランダム選択の乱数シードとして適している。

- 将来生成されるブロックハッシュを知ったり，望みの値に設定したりすることが，非常に困難（数千万円の費用がかかるため，実質上不可能）
  → ランダム選択の結果を締切り前に予想したり，不正に操作したりすることが（実質上）不可能
- 一度ブロックハッシュが決まれば，その値を誰でも知ることが可能
  → ランダム選択のプロセスに不正や誤りが無いことを，ブロックハッシュ値から計算した乱数シードを用いて，誰でも後から検証可能

2　添付の Python スクリプトでは，「ビットコインの block height 629530-629534 の 5 個のブロックハッシュの和」を乱数シードとした例を示している。

- ブロックハッシュ値は，https://explorer.btc.com/btc/block/[block height] から取得可能である（例えば，block height 629530 であれば https://explorer.btc.com/btc/block/629530）。
- Python スクリプトを実行すると下記の結果が出力される。乱数シードが同じ限り，何度実行しても同じ結果が得られる。

```
Entries: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 14, 15, 16, 17,
18, 19, 20, 21, 23, 25, 28, 29, 31, 33] (26 in total)
Random seed:
33289223846857809242230034440972413870415546845345171 40

Result
Selected entries: [1, 2, 3, 5, 7, 8, 9, 10, 11, 16, 17, 18, 21, 23,
25]
```

ランダム選択を行う Python コード
https://ideone.com/IrZVK5

```
import random
import platform

assert platform.python_version()[0:3] == "3.7", "Python version 3.7 must be used."

# The number of selections
NUM_SELECTED = 15
# Entry numbers that passed the screening process (example is shown)
ENTRIES = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 14, 15, 16, 17, 18, 19, 20, 21, 23,
25, 28, 29, 31, 33]
assert NUM_SELECTED < len(ENTRIES), "Selection must happen"

# Block hashes from certain block heights that were previously announced:
# Below is the example by block heights 629530-629534.
hashes = [
    0x00000000000000000006F349AA480F67A2B603496DA07FD0F566680293B2D3E4,
    0x00000000000000000000E4BF1CA971D88B29D31B84751AE6BDF8F2F5F25E5D99E,
    0x00000000000000000003A91B8D6D37940269AE8DE9219176DCD6BA448CE0AC75,
    0x00000000000000000000137A2AC232E19D2163A4A28B2F1F49CCD35052579451E,
    0x00000000000000000008A17371C0F62112227C28B83DD88C5218CAD648484E7F,
]

seed = sum(hashes)
random.seed(seed)
print("Entries:", ENTRIES, "(%d in total)" % len(ENTRIES))
print("Random seed: %d" % seed)
print()

number_order = sorted(random.sample(ENTRIES, len(ENTRIES))[0:NUM_SELECTED])

print("Result")
print("Selected entries:", number_order)
```