DNA 전사 과정에서의 RNA 염기배열 돌연변이의 발생 확률에 관한 실험

Huere

전사는 RNA 중합 효소가 전사 개시 앞 부위에 있는 프로모터[1]에 결합하여 시작된다. 프로모터에 결합한 RNA 중합 효소는 DNA 이중 나선을 풀고, 이 중 한 가닥을 주형으로 사용하여 5‘ -> 3’ 방향으로 뉴클레오타이드를 연결하여 상보적인 RNA 가닥을 합성한다. 이때 DNA 복제와는 달리 DNA의 A에 상보적인 염기로 U를 사용하며, 합성된 RNA는 주형 가닥에서 떨어져 나오고, 합성이 완료된 부위의 DNA 가닥은 다시 이중 나선을 형성한다. RNA 중합 효소가 전사 종결 부위에 도달하면 합성된 RNA가 방출되고, RNA 중합 효소가 DNA에서 떨어져 나온다.

RNA는 전령 RNA(mRNA), 리보솜 RNA(rRNA), 운반 RNA(tRNA)의 세 종류가 있다. 이 가운데 mRNA는 단백질 합성 정보를 전달하는 역할을 한다. rRNA는 단백질 합성 기구인 리보솜을 구성하는 주요 성분으로 단백질 합성을 촉진하는 핵심적인 역할을 하며, tRNA는 단백질 합성 과정에서 아미노산을 리보솜으로 운반하는 역할을 한다.

점 돌연변이는 하나의 뉴클레오타이드가 변환되어 나타나는 돌연변이로, 앞서 설명한 전사 과정에서 모종의 이유로 전사가 잘못된다면 돌연변이가 일어난다. 대부분 같은 퓨린 계열인 A와 G의 잘못된 전사, 또는 같은 피리미딘 계열인 C과 T의 잘못된 전사로 인해 일어난다. 이러한 점 돌연변이는 코돈의 정보를 파괴하거나 변형시킨다. 이로 인해 일어나는 몇가지 문제가 있는데, 불현성 돌연변이, 미스센스 돌연변이, 넌센스 돌연변이를 일으킨다.

불현성 돌연변이는 다른 말로 침묵 돌연변이라고도 하는데, 단백질의 기능에 영향을 주지 않는다. 따라서 단백질 자체는 돌연변이가 없는 정상 단백질이 만들어진다. 이것이 가능한 이유는 64개의 코돈[2]이 20개의 아미노산과 종결 코돈만을 암호화하기 때문이다. 하지만 달라진 코돈이 단백질 합성 수에는 영향을 미칠 수 있다.

미스센스 돌연변이는 코돈을 바꿔서 다른 단백질을 만든다. 미스센스 돌연변이는 보존적 돌연변이와 비보존적 돌연변이로 나뉘는데, 보존적 돌연변이는 아미노산을 바꾸지만 친수성, 소수성, 친유성 등 성질은 바꾸지 않는다. 가끔은 하나의 아미노산 변화가 생물에 전체적으로 해롭지 않을 수 있다. 대부분의 단백질은 그 기능이 변하기 전에 한두 개 정도 점 돌연변이가 있다. 반면에 비보존적 돌연변이는 다른 성질을 띄게 한다. 돌연변이가 일어난 단백질은 기능을 잃어 질병을 유발할 수 있다. 그 예로 낫

1) 유전자 DNA에 존재하는 조절 부위로, RNA 중합 효소가 결합하여 전사가 시작되는 곳.
2) mRNA에 있는 하나의 아미노산을 지정하는 유전부호를 코돈이라고 부른다.

모양 적혈구 빈혈증이 있다. 베타 헤모글로빈 유전자에서 점돌연변이가 한 번 일어나 GAG 코돈이 GUG[3]로 바뀌고 이에 따라 글루탐산[4]이 발린[5]으로 바뀌면 발병하게 된다. 반대로 기능을 잃는 것 대신 얻을 수도 있다. 예를 들어 BRAF[6] 유전자에서 발린이 글루탐산으로 바뀌면 RAF 단백질이 활성화되는데, 이는 암세포에서 증식 신호의 무제한 발생을 유도한다. 이로 인해 종양이 발생한다.

넌센스 돌연변이는 염기가 치환되어버려 대응되는 아미노산이 없어 코돈이 무의미로 변하고 단백질 합성이 변이 부위에서 정지되는 돌연변이이다. 이는 종결획득과 개시상실로 나뉜다. 종결획득은 너무 이른 지점에 종결 코돈이 만들어져 번역이 미완성으로 끝나는 돌연변이이다. 이때 사라진 아미노산의 개수가 단백질의 기능과 정상 작동 여부를 결정한다. 종결상실은 기존 종결 코돈에 발생하는 돌연변이로 비정상적으로 긴 C말단[7]을 생성한다.

나는 이 점돌연변이가 어떤 염기에서 주로 일어날지 통계적으로 확인해보고 싶었다. 하지만 관련된 실험이 없었고, 현실적으로 이런 실험을 실제로 할 수는 없기 때문에 실험을 직접 설계해야 했다. 소프트웨어의 list 자료형을 사용하면 DNA의 염기서열과 비슷한 환경을 구현할 수 있다. 이 환경을 위해 변인 통제 단계에서 조작변인을 몇 가지 조작해야한다.

1. 염기의 개수에 따른 확률에 대한 영향을 배제하기 위해 RNA의 초기 염기서열의 A,G,C,T의 비율은 같아야 한다.

2. 믿을 수 있고 보다 정확한 결과를 위해 실험의 Epoch[8]는 10000번 이상으로 제한한다. 자연적인 돌연변이는 100만 번에 한 번 발생하는데, 이를 모두 시뮬레이션 하기에는 물리적 시간의 어려움이 있기 때문이다.

3. A는 같은 계열인 G로 변화할 확률이 더 커야한다. 같은 이유로 G는 A로, C는 T로, T는 C로 변화할 확률이 더 커야한다. 이때 증가되는 확률은 20%로 한다.

4. A가 G로 변화할 경우 자연의 법칙에 위배되기 때문에 G로 변화할 경우 G를 U로 대체한다.

---

3) RNA의 전사 과정에서 A는 U로 대응된다.
4) 글루탐산은 20가지 단백질성 아미노산 가운데 하나이며 코돈은 GAA, GAG이다.
5) 발린은 알파-아미노산의 하나로, 필수 아미노산 중 하나이다.
6) B-Raf 단백질을 암호화하는 인간 유전자로써, 세포의 성장과 분화, 사멸에 관여하는 MAPK의 중요 인자이다. MAPK는 유사 분열, 삼투 스트레스, 열 충격 등 다양한 자극에 관한 세포 반응을 유도하는 데 관여하는 효소이다.
7) 폴리펩타이드와 같은 아미노산 사슬의 끝부분이다.
8) 코드의 루프 횟수

5. 돌연변이를 일으키는 과정에서 염기의 손실은 없는 걸로 가정한다. 염기의 개수가 달라지는 경우 확률을 정확하게 조사할 수 없기 때문이다.

6. 실험 기간을 줄이기 위해 한 Epoch당 반드시 한 개 이상의 염기에서 돌연변이가 일어난다고 가정한다.

7. 코돈을 구현하는 것은 코드단위에서 인식 문제가 있기 때문에 단일 염기의 돌연변이만을 가정하고, 돌연변이가 개체에 어떤 영향을 미칠지에 대한 예측은 배제한다.

상기 7개의 조작변인을 소프트웨어에 적용하고 코드를 작성했다. 초기 염기는 'AGCTAGTCGACTACGTAGCATGTCAGCTGACATGTCGACT'로 제시했고, Epoch는 10000(1만 번)으로 설정했다. 통계에 따르면 현실에서의 100만 * 1만(100억) 번의 분화를 실험한 셈이다. 실험 순서는 아래와 같다.

1. 제시된 염기서열의 인덱스 값을 불러온다.

2. 무작위 염기를 무작위 염기로 변화시킨다.

4. 1번과 2번을 Epoch만큼 반복한다.

5. 각 염기의 비율을 구한다.

6. 초기 염기와 비교해 변화한 염기의 각각의 비율을 구한다.

7. 실험 결과를 분석한다.

코드는 약 164줄로, DNA 구성 오픈소스를 첨삭해서 작성했다. 기존 코드와는 크게 다른 부분이 많아 새로운 코드가 작성되었다. 이 코드를 모두 수록하려면 지면이 크게 소요가 되니 핵심 코드만 수록하겠다(핵심 코드는 그림1과 그림2를 참고). 각 염기가 다른 염기로 변화할 확률은 모두 50%이며, 같은 계열의 염기로 분화할 확률만 20% 높게 설정했다. 각 인덱스의 염기를 지웠다가 무작위 염기를 다시 삽입하는 방식으로 돌연변이를 구현했다. 결과는 그림3과 같다.
initial rna는 초기에 제시한 염기서열, changed rna는 실험 결과로 나타난 염기서열, 아래의 알파벳은 염기이며 그 옆의 숫자는 돌연변이를 일으켜 기존의 염기를 대체한 염기의 비율이다. everage는 Epoch를 수행하는 도중 일어난 돌연변이의 횟수이다. total percentage는 100%가 아닐 때 실험이 실패한 것인데, 이를 확인하려고 삽입했다. 99.999%는 계산의 오차이다. 실패라고 할 정도로 큰 오차가 아니므로 무시

한다.

실험 결과 염기는 U와 G로 변화할 확률이 약 8% 더 높은 것으로 확인됐다. 또, C, T, A 염기로 변화할 확률은 15%로 비슷했다. U와 G는 둘 다 A와 관련되어 있고 각 염기는 A로도 변화할 확률이 있어 U와 G가 더 높게 측정된 것으로 분석된다. 이는 실제로도 적용되는 법칙이므로 합리적인 실험결과이다. 앞서 서술한 낫모양 적혈구 빈혈증이 GAG에서 GUG로 변화하는 돌연변이이다. 이 돌연변이는 소아기 시기에 발현을 시작하고 열대 지방의 흑인에게 흔해서 대표적인 돌연변이인데, 염기가 U로 변화할 확률이 더 높아서일 수도 있다는 합리적인 추론이 가능해진다. 또한, 종결코돈은 UAA, UAG, UGA 등 U와 G의 비율이 높은데, 앞서 서술한 넌센스 돌연변이는 이 종결코돈이나 코돈의 염기가 바뀌면서 생성된다. 야생형 기능이 손상된 잘린 p53 단백질이 발현되게 하는 것도 넌센스 돌연변이인데, 암을 가장 빈번하게 발생시키는 TP53유전자의 약 10%정도는 넌센스 돌연변이 때문에 일어난다[9]. 10%라는 비율은 크진 않지만, 암의 모든 TP53의 돌연변이의 10%라고 생각하면 다른 돌연변이에 비해 적은 것이 아니다. 이는 실험결과의 영향이 어느 정도 있다는 추론도 가능하다.

하지만 이 실험은 모든 요인을 전부 고려하지 않았을 뿐 아니라 고정적인 로직으로 구현되는 시뮬레이션이기 때문에 정확한 결과도 아닐뿐더러 심지어 실험 결과가 실제와는 크게 다를 수도 있다. 하지만 큰 조건들은 모두 고려했고 실험 횟수도 1만 번으로 현실의 100억 번 정도를 실험했으니 비교적 정확한 결과라고 할 수 있다. 추후에 더 고도화된 로직으로 현실의 조건을 고려해서 더 나은 시뮬레이션을 제작하게 된다면 더 정확하고 현실적인 결과가 도출될 것이다. 또한 추후에 느리고 적지만 정확한 실제 데이터를 소수 구축하고 이를 토대로 빠르고 많지만 비교적 부정확한 시뮬레이션 결과를 축적해서 대조한다면 실제 돌연변이가 어디서 나타날지에 대한 경향이 계산되기 때문에 돌연변이가 어디서 나타날지에 대한 예측을 도출할 수도 있을 것이다.

9) p53 돌연변이를 타겟하는 효과적인 암 치료 / 2019.08.21. / 도민재(KAIST 화학과) / https://www.ibric.org/myboard/read.php?Board=report&id=3292

```
if dna_sequence[ch_ID] == 'A':
    ch_ID1 = ch_ID - 1

    DNA_part1 = dna_sequence[0:ch_ID1]
    DNA_part2 = dna_sequence[ch_ID:len(dna_sequence)]
    if random.randrange(0, 11) >= 7:
        dna_sequence = DNA_part1 + 'U' + DNA_part2
    else :
        dna_sequence = DNA_part1 + changeInto() + DNA_part2

elif dna_sequence[ch_ID] == 'G':
    ch_ID1 = ch_ID - 1

    DNA_part1 = dna_sequence[0:ch_ID1]
    DNA_part2 = dna_sequence[ch_ID:len(dna_sequence)]
    if random.randrange(0, 11) >= 7:
        dna_sequence = DNA_part1 + 'A' + DNA_part2
    else :
        dna_sequence = DNA_part1 + changeInto() + DNA_part2

elif dna_sequence[ch_ID] == 'C':
    ch_ID1 = ch_ID - 1

    DNA_part1 = dna_sequence[0:ch_ID1]
    DNA_part2 = dna_sequence[ch_ID:len(dna_sequence)]
    if random.randrange(0, 11) >= 7:
        dna_sequence = DNA_part1 + 'T' + DNA_part2
    else :
        dna_sequence = DNA_part1 + changeInto() + DNA_part2
```

그림 1

```
elif dna_sequence[ch_ID] == 'T':
    ch_ID1 = ch_ID - 1

    DNA_part1 = dna_sequence[0:ch_ID1]
    DNA_part2 = dna_sequence[ch_ID:len(dna_sequence)]
    if random.randrange(0, 11) >= 7:
        dna_sequence = DNA_part1 + 'G' + DNA_part2
    else :
        dna_sequence = DNA_part1 + changeInto() + DNA_part2

elif dna_sequence[ch_ID] == 'U':
    ch_ID1 = ch_ID - 1

    DNA_part1 = dna_sequence[0:ch_ID1]
    DNA_part2 = dna_sequence[ch_ID:len(dna_sequence)]
    if random.randrange(0, 11) >= 7:
        dna_sequence = DNA_part1 + 'A' + DNA_part2
    else :
        dna_sequence = DNA_part1 + changeInto() + DNA_part2
```

그림 2

```
initial rna : AGCTAGTCGACTACGTAGCATGTCAGCTGACATGTCGACT
changed rna : TGGTTTCUGUGAGAGTGTCTUUGTTUTAGTATCTGGAAGT
everage : 160880
U : 27.6
G : 22.4
C : 14.8
T : 14.8
A : 15.0
total percentage : 99.99999999999999
```

그림 3