(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 106480189 B
(45)授权公告日 2018.11.09

(21)申请号 201610902786.5

(22)申请日 2016.10.18

(65)同一申请的已公布的文献号
申请公布号 CN 106480189 A

(43)申请公布日 2017.03.08

(73)专利权人 中国水产科学研究院黄海水产研究所
地址 266003 山东省青岛市市南区南京路106号

(72)发明人 陈松林 刘洋 刘峰 李仰真 邵长伟 王磊 王娜 周茜 刘寿堂 翟介明 郑卫卫 张英平 孙河军 杨英明

(74)专利代理机构 青岛海昊知识产权事务所有限公司 37201
代理人 曾庆国

(51)Int.Cl.
C12Q 1/6888(2018.01)
A01K 61/13(2017.01)
G06F 19/18(2011.01)

(56)对比文件
CN 104313135 A,2015.01.28,
CN 103882144 A,2014.06.25,
CN 101911918 A,2010.12.15,

审查员 王溯铭

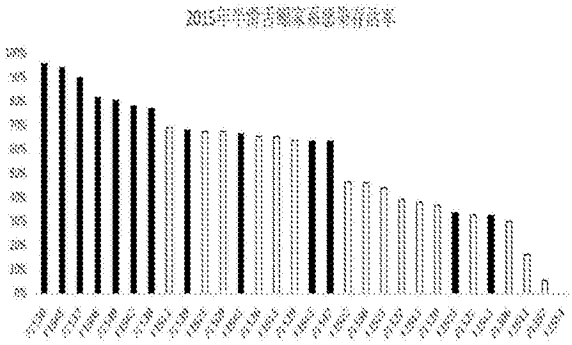
权利要求书2页 说明书34页 附图7页

(54)发明名称

一种基于全基因组选择的鱼类抗病良种培育方法

(57)摘要

本发明的目的是建立一种基于全基因组选择的鱼类抗病良种培育方法，弥补传统育种技术的不足，为鱼类抗病高产优质良种培育提供分子育种方法，解决鱼类养殖业中缺乏全基因组选择方法的问题，为鱼类养殖业提供抗病优良品种选育的技术手段，实现鱼类育种技术的更新换代，推动鱼类种业快速发展。本发明的方法获得的抗病苗种的抗病力明显提高。实验结果表明通过全基因组选择方法培育的抗病苗种的感染存活率和养殖存活率比对照组高出20%-30%；通过这种方法可以快速、高效培育出鱼类抗病优良品种，提高鱼类的抗病能力和养殖成活率，在鱼类养殖业中具有重大的应用价值和广阔的推广前景。



CN 106480189 B

1.一种基于全基因组选择的半滑舌鳎、牙鲆鱼类抗病良种培育方法，其特征在于，所述的方法包括如下步骤：

1)半滑舌鳎、牙鲆鱼类抗病参考群体的建立及表型性状测定

当建立的鱼类家系鱼苗生长达到全长8-15厘米时，采用病原微生物对鱼类家系鱼苗进行病原菌人工感染，在病原菌感染发病后不同时间收集各家系死亡的鱼苗，记录鱼苗死亡时间及全长、体重、体宽表型数据，同时采集感染后存活的个体并记录全长、体重、体宽数据，将采集的存活个体与死亡个体的数据作为评价鱼体抗病力的表型参数，并按照死亡率和死亡时间，在各家系中选取实验个体，得到一组可反应整个实验群体抗病性状的代表性个体的组合作为参考群体；采用ASReml软件的动物模型计算参考群体个体的估计育种值EBV和群体育种均值，并转化为相对育种值RBV，作为基因组选择计算中参考群体的表型；

2)参考群体的全基因组重测序及基因型采集与分析处理

提取采集的参考群体鱼苗的基因组DNA样品建库测序，建库类型为DNA-350bp，测序策略为HiSeqPE150，对测序结果首先根据测序质量，过滤掉含有接头序列和低质量碱基的测序所得reads，比对参考基因组，检测参考基因组中对应的SNP位点并calling，整理生成.vcf文件；使用plink对.vcf文件进行读取、合并个体数据，随后按照染色体进行分割和质量控制、使用beagle进行数据填充，再将所有用到的染色体上的SNP位点数据合并，转换生成以个体为单位的.csv格式基因型文件，用于基因组选择的计算；

所述的参考基因组为半滑舌鳎和牙鲆的全基因组测序结果，其中半滑舌鳎全基因组测序结果，其在GenBank中的ID为PRJNA73987；牙鲆的全基因组测序结果在GenBank中的ID为PRJNA73673；

3)参考群体的全基因组选择计算和SNP位点效应分析

以参考群体的相对育种值RBV为表型，以参考群体测序所得SNP位点数据为基因型，进行基因组选择计算，使用的计算方法为Bayes Cπ，所使用的计算工具为R-package BGLR，计算后获得每个SNP位点与抗病表型的相关性数值，整理输出.txt文件，作为各个SNP位点的遗传效应值，用于全基因组选择中的基因组育种值GEBV的估计；

4)候选群体的建立与全基因组重测序

采集候选群体鱼的鳍条，提取鳍条基因组DNA，进行基因组文库构建和全基因组重测序，处理分析SNP位点数据，在缺失表型值的情况下，进行基因组选择计算；在参考群体基因组选择计算完成后，通过计算出的SNP位点的效应值，结合候选群体个体各SNP位点基因型，得到候选群体基因组估计育种值；

5)参考群体和候选群体的个体基因组估计育种值GEBV分析

根据获得的参考群体的每个SNP位点的遗传效应和不同个体各个SNP位点的基因型，计算得出参考群体的个体基因组估计育种值GEBV；在计算结束后生成并输出.txt记录参考群体个体的GEBV；根据参考群体个体的GEBV，初步筛选出参考群体中抗病力强的个体，以及这些个体所在的家系；根据参考群体个体的SNP位点效应和候选群体的SNP基因型，进一步计算获得候选群体个体的基因组估计育种值；根据候选群体的GEBV计算结果，计算GEBV值与实际感染存活率的相关系数，进而验证基因组选择计算结果的准确性；

6)半滑舌鳎、牙鲆鱼类抗病苗种的培育

根据全基因组选择计算得出的候选群体基因组估计育种值的大小，在表型缺失的情况

下，对候选群体的抗病力进行评估和排序；选择GEBV高的候选群体个体作为亲鱼，进行抗病苗种的繁育；繁育所得的子代苗种的抗病力显著提高。

2.如权利要求1所述的方法，其特征在于，所述步骤1)中转化为相对育种值，是使用R语言环境下的R-package asreml计算获得的。

3.如权利要求1所述的方法，其特征在于，所述步骤4)中所述的候选群体是指没有进行病原菌感染、没有抗病性状表型的半滑舌鳎、牙鲆鱼类家系个体。

一种基于全基因组选择的鱼类抗病良种培育方法

技术领域

[0001] 本发明属于水产遗传育种技术领域，具体涉及一种基于全基因组选择的鱼类抗病良种培育方法。

背景技术

[0002] 鱼类养殖业是我国水产养殖业的支柱产业，2014年鱼类养殖产量2721.94万吨，占整个水产养殖产量的57.3％。养殖鱼类是我国食物蛋白的重要来源。

[0003] 然而，随着鱼类养殖业的迅速发展，缺乏优良品种、养殖种类种质退化现象突出；养殖规模的扩大、集约化程度的提高以及养殖环境的恶化而导致的水产养殖病害发生频繁，养殖产品药残突出等问题严重制约着鱼类养殖业的可持续发展。仅就鱼类而言，由于高密度养殖形成的免疫压抑，导致养殖鱼类的抗病力下降；由于对鱼类的免疫抗病机理及抗病力的分子遗传基础缺乏了解，难以从分子水平提出防治鱼类病害发生及发展的技术途径；由于缺乏抗病功能基因和抗病分子标记，难以进行抗病优良品种培育，养殖生产只能依靠抗病力差的野生或人工繁殖多代的苗种进行，因而流行性病害在鱼类养殖中频繁发生。据不完全统计，我国每年因病害给鱼类养殖业造成的直接经济损失达100亿元之巨。病害已成为制约我国鱼类养殖业可持续发展的瓶颈。目前危害养殖鱼类的主要病害是细菌性和病毒性疾病以及寄生虫类疾病。其中危害较大的3种细菌性疾病分别是爱德华氏菌病、链球菌病和弧菌病。抗菌素类药物或者疫苗等防病措施虽然有一定效果，但无法从根本上解决水产养殖中的病害问题。且抗菌素类药物具有容易在鱼体内积累，降低养殖鱼类商品质量，对消费者的健康具有潜在危害，容易使病原菌产生抗药性以及严重污染养殖环境等问题，因而在水产养殖业中的应用越来越受到限制。同时抗菌素的使用也不能满足人们日益增长的对无药物残留的绿色水产品的需求。因此，鱼类抗病良种选育是我国水产领域急需攻克的重大课题之一。

[0004] 迄今为止，鱼类良种选育主要是基于表型性状的选育，包括群体选育、家系选育、杂交选育和BLUP选育等，都是根据体长、体重等容易测量的表型值计算出来的育种值进行选择。分子标记出现以后，则是通过定位与重要经济性状相关联的分子标记从而对经济性状进行基因型选择，但传统的分子标记辅助选育所用的分子标记数量非常有限，对单基因性状或质量性状的选择效果不错，但对于多基因决定的数量性状的选择效果则不太好。由于抗病性状是由多个基因控制的数量性状，难以测量，选择准确性相当低，所以对抗病良种的选育一直进展缓慢，限制了鱼类抗病新品种的培育。迫切需要一种新的育种技术或手段来攻克这一难题。因此，本发明建立了一种基于全基因组选择的鱼类抗病良种培育方法，旨在为鱼类抗病良种培育提供一种新的分子育种技术。

发明内容

[0005] 本发明的目的是建立一种基于全基因组选择的鱼类抗病良种培育方法，弥补传统育种技术的不足，为鱼类抗病高产优质良种培育提供分子育种方法，解决鱼类养殖业中缺

乏全基因组选择方法的问题，为鱼类养殖业提供抗病优良品种选育的技术手段，实现鱼类育种技术的更新换代，推动鱼类种业快速发展。

[0006] 本发明的基于全基因组选择的鱼类抗病良种培育方法，主要包括如下步骤：

[0007] 1)鱼类抗病参考群体的建立及表型性状测定

[0008] 当建立的鱼类家系鱼苗生长达到全长8-15厘米时，采用病原微生物对鱼类家系鱼苗进行病原菌人工感染，在病原菌感染发病后不同时间收集各家系死亡的鱼苗，记录鱼苗死亡时间及全长、体重、体宽等表型数据，同时采集感染后存活的个体并记录全长、体重、体宽数据，将采集的存活个体与死亡个体的数据作为评价鱼体抗病力的表型参数，并按照死亡率和死亡时间，在各家系中选取实验个体，得到一组可反应整个实验群体抗病性状的代表性个体的组合作为参考群体；采用动物模型计算参考群体个体的估计育种值(EBV)和群体育种均值，并转化为相对育种值(RBV)，作为基因组选择计算中参考群体的表型；

[0009] 所述转化为相对育种值，是使用R语言环境下的R-package asreml计算获得的；

[0010] 2)参考群体的全基因组重测序及基因型采集与分析处理

[0011] 提取采集的参考群体鱼苗的基因组DNA样品建库测序，建库类型为DNA-350bp，测序策略为HiSeqPE150，对测序结果首先根据测序质量，过滤掉含有接头序列和低质量碱基较多的测序所得reads，比对参考基因组，参考基因组为半滑舌鳎和牙鲆的全基因组测序结果(GenBank ID:PRJNA73987;PRJNA73673)，检测参考基因组中对应的SNP位点并calling，整理生成.vcf文件；使用plink对.vcf文件进行读取、合并个体数据，随后按照染色体进行分割和质量控制、使用beagle进行数据填充，再将所有用到的染色体上的SNP位点数据合并，转换生成以个体为单位的.csv格式基因型文件，用于基因组选择的计算；

[0012] 3)参考群体的全基因组选择计算和SNP位点效应分析

[0013] 以参考群体的相对育种值(RBV)为表型，以参考群体测序所得SNP位点数据为基因型，进行基因组选择计算，使用的计算方法为Bayes Cπ，所使用的计算工具为R-package BGLR，计算后获得每个SNP位点与抗病表型的相关性数值，整理输出.txt文件，作为各个SNP位点的遗传效应值，用于全基因组选择中的基因组育种值(GEBV)的估计；

[0014] 4)候选群体的建立与全基因组重测序

[0015] 采集候选群体鱼的鳍条，提取鳍条基因组DNA，进行基因组文库构建和全基因组重测序，处理分析SNP位点数据，在缺失表型值的情况下，进行基因组选择计算；在参考群体基因组选择计算完成后，通过计算出的SNP位点的效应值，结合候选群体个体各SNP位点基因型，得到候选群体基因组估计育种值；

[0016] 所述的候选群体是指没有进行病原菌感染、没有抗病性状表型的鱼类家系个体，从不同家系中选择一部分不知抗病表型性状的鱼类家系个体或者在基因组选择实际应用与鱼类良种培育时，选取用于繁育的亲鱼作为候选群体；

[0017] 5)参考群体和候选群体的个体基因组估计育种值(GEBV)分析

[0018] 根据获得的参考群体的每个SNP位点的遗传效应和不同个体各个SNP位点的基因型，计算得出参考群体的个体基因组估计育种值(GEBV)；在计算结束后生成并输出.txt记录参考群体个体的GEBV；根据参考群体个体的GEBV，初步筛选出参考群体中抗病力较强的个体，以及这些个体所在的家系；根据参考群体个体的SNP位点效应和候选群体的SNP基因型，进一步计算获得候选群体个体的基因组估计育种值；根据候选群体的GEBV计算结果，计

算GEBV值与实际感染存活率的相关系数，进而验证基因组选择计算结果的准确性。

[0019] 6)鱼类抗病苗种的培育

[0020] 根据全基因组选择计算得出的候选群体基因组估计育种值的大小，在表型缺失的情况下，对候选群体的抗病力进行评估和排序；选择GEBV高的候选群体个体作为亲鱼，进行抗病苗种的繁育；繁育所得的子代苗种的抗病力显著提高；

[0021] 本发明的方法获得的子代苗种在后续进行的感染实验中，存活率显著高于对照组，结果表明基因组选择候选群体的后代苗种的感染存活率比对照组高出20％-30％；通过这种方法可以快速、高效培育出鱼类抗病优良品种，提高鱼类的抗病能力和养殖成活率，在鱼类养殖业中具有广泛的应用价值和推广前景。

附图说明

[0022] 图1：牙鲆全基因组重测序样品SNP位点在各染色体上的分布。横坐标chr1-chr24表示24条染色体，纵坐标表示染色体上SNP位点的数目，最小值为chr16的5223，最大值为chr1的20971。

[0023] 图2：半滑舌鳎全基因组重测序样品SNP位点在各染色体上的分布。横坐标chr1-chr20表示20条常染色体，纵坐标表示染色体上SNP位点的数目，最小值为chr16的51595，最大值为chr1的200654。

[0024] 图3：牙鲆基因组选择样品计算出的397215个SNP位点的效应值。横坐标表示各SNP位点按照在基因组上位置排列，纵坐标表示SNP位点效应值，大小在1.18E-16至3.73E-4之间。

[0025] 图4：半滑舌鳎基因组选择样品计算出的17529015个SNP位点的效应值。横坐标表示各SNP位点按照在基因组上位置排列，纵坐标表示SNP位点效应值，大小在3.37E-19至3.83E-5之间。

[0026] 图5：牙鲆参考群体GEBV，以牙鲆参考群体相对育种值(RBV)为表型，计算出的参考群体GEBV，大小在64.2-130.1之间。横坐标表示牙鲆参考群体个体，纵坐标为GEBV。

[0027] 图6：牙鲆候选群体中母本的子代存活率与GEBV的比较，将子代存活率与GEBV分别排序，比较二者相关性，相关系数为0.81。横坐标为候选群体个体，纵坐标为排序名次。

[0028] 图7：牙鲆候选群体中父本的子代存活率与GEBV的比较，将子代存活率与GEBV分别排序，比较二者相关性，相关系数为0.89。横坐标为候选群体个体，纵坐标为排序名次。

[0029] 图8：半滑舌鳎参考群体GEBV，以半滑舌鳎参考群体相对育种值(RBV)为表型，计算出的参考群体GEBV，大小在18.3-156.9之间。横坐标表示半滑舌鳎参考群体个体，纵坐标为GEBV。

[0030] 图9：牙鲆苗种迟缓爱德华氏菌感染后存活率比较。通过基因组选择方法培育出牙鲆抗病苗种(命名为鲆优2号)，用迟缓爱德华氏菌感染鲆优2号和对照组，计算感染后存活率，鲆优2号存活率为74％，对照组存活率为44％。

[0031] 图10：牙鲆苗种养殖存活率比较。通过基因组选择方法培育出牙鲆抗病苗种(命名为鲆优2号)，其养殖存活率为64％；而对照组养殖存活率为39％。

[0032] 图11：牙鲆苗种养殖日增重比较。通过基因组选择方法培育出牙鲆抗病苗种(命名为鲆优2号)，进行了鲆优2号和对照组的生长对比实验，计算养殖日增重，鲆优2号组0.82g/

天，对照组为0.66g/天。

[0033] 图12：2015年半滑舌鳎家系哈维氏弧菌感染存活率，黑色柱形代表基因组选择候选群体个体为父本的家系，白色柱形代表未经基因组选择的普通舌鳎为父本的家系，可见基因组选择家系苗种的存活率大多较高。

[0034] 图13：2015年半滑舌鳎群体哈维氏弧菌感染存活率，将半滑舌鳎家系按照父本为基因组选择候选群体个体或普通个体分为两个群体，计算存活率，基因组选择出的苗种的存活率为67.6％，普通群体存活率为43％。

具体实施方式

[0035] 下面以牙鲆和半滑舌鳎为例，结合附图对基于全基因组选择的鱼类抗病良种培育方法进行详细叙述：

[0036] 一、鱼类抗病参考群体的建立及表型性状测定

[0037] 当建立的鱼类家系鱼苗生长达到全长8-15厘米时，采用病原微生物对鱼类家系鱼苗进行病原菌人工感染，在病原菌感染发病后不同时间收集各家系死亡的鱼苗，记录鱼苗死亡时间及全长、体重、体宽等表型数据，同时采集感染后存活的个体并记录全长、体重、体宽数据，将采集的存活个体与死亡个体数据作为评价鱼体抗病力的表型参数，并按照死亡率和死亡时间，在各家系中选取实验个体，得到一组可反应整个实验群体抗病性状的代表性个体的组合作为参考群体；采用动物模型计算参考群体个体的估计育种值(EBV)和群体育种均值，并转化为相对育种值(RBV)，作为基因组选择计算中参考群体的表型；具体计算使用的是R语言环境下的R-package asreml。下面以牙鲆和半滑舌鳎为例进行详细叙述。

[0038] (一)、牙鲆抗迟缓爱德华氏菌病参考群体的建立及表型性状测定

[0039] 1.牙鲆迟缓爱德华氏菌感染实验

[0040] 自2003年起就开始了牙鲆选择育种，首先通过自然选择和人工感染途径以及不同地理群体的收集和选育，得到具有不同性状的牙鲆鱼苗，培育至性成熟后，得到牙鲆选育亲鱼第一代，包括：抗病群体(RS)、日本群体(JS)、韩国群体(KS)、黄海群体(YS)。以这些选育亲鱼作为亲本，建立牙鲆家系。

[0041] 2007年以RS、JS、YS等为基础群体，建立了F1代63个牙鲆家系，通过对其中59个家系进行鳗弧菌感染实验，筛选出4个抗病家系，鳗弧菌感染后存活率达到50％以上。

[0042] 依据2007年63个家系后裔的鉴定结果，2008年选择不同群体的优良亲本，根据双列杂交法建立30个家系，通过对63个F1家系中的30个家系进行鳗弧菌感染实验，筛选出5个抗鳗弧菌家系。

[0043] 利用RS、JS、YS以及2007年选留的优良家系为亲本，2009年共建成家系43个，选取其中33个家系进行鳗弧菌感染实验，其中F1家系26个，回交家系1个，雌核发育一代家系2个，F2家系4个，筛选出抗鳗弧菌家系6个。

[0044] 2010年，以2007年抗鳗弧菌感染的牙鲆(RS)与日本引进后选育的牙鲆群体(JS)进行交配得到的养殖成活率高、生长较快的杂交群体的雌鱼作为母本，与韩国引进选育牙鲆群体(KS)作为父本进行杂交，得到的三杂交后代即为牙鲆“鲆优1号”。

[0045] 2012年，以RS、JS、YS以及2007、2009年选留的优良家系为亲本，建成家系65个，选取43个家系进行鳗弧菌感染实验，其中31个F1家系，5个F2家系，6个F3家系和1个雌核发育

二代家系，筛选出抗鳗弧菌家系8个。

[0046] 2013年，以2007年建立的生长快、成活率高的优良家系，韩国群体(KS)及其自交后代，2009年在抗鳗弧菌感染、抗LCDV、养殖成活率和日增重率表现优良的家系为亲本，建立牙鲆家系56个，其中32个家系用于迟缓爱德华氏菌人工感染实验，筛选到抗迟缓爱德华氏菌家系6个。细菌感染实验按照已报道的方法进行。

[0047] 2014年，以韩国群体，2007、2009、2010年优良家系为亲本，建立牙鲆家系47个，对其中的39个家系进行了感染实验，得到抗迟缓爱德华氏菌家系7个。

[0048] 2015年4月开始，以本实验室历年来通过生长和抗病性能分析筛选出的优良牙鲆作为亲本，建立牙鲆家系56个。选取46个家系进行迟缓爱德华氏菌感染实验，包括10个F3家系、36个F4家系，其中全同胞家系9个，半同胞家系31个，其他组合家系6个。筛选到5个抗迟缓爱德华氏菌牙鲆家系。

[0049] 2013年-2015年，连续3年在牙鲆全长达到8-15cm，符合实验规格时，进行牙鲆迟缓爱德华氏菌感染实验，收集了感染个体在感染后的存活时间、全长、体宽、体重等表型数据，同时采集了死亡鱼苗和存活鱼苗的鳍条样品(表1)，用于基因组DNA的提取，基因组选择实验参考群体即选自这些感染样品。

[0050] 表1 各年份迟缓爱德华氏菌感染牙鲆

[0051]

感染年份	家系数	个体数	存活率
2013	34	4577	31.2%
2014	40	5942	20.3%
2015	48	6919	48.9%

[0052] 2.牙鲆抗迟缓爱德华氏菌参考群体表型性状的分析

[0053] 使用分析软件ASReml-R，对参考群体表型进行分析，选用软件中的动物模型计算2013、2014、2015年感染实验的遗传力和每个个体的估计育种值(EBV)。为使3年的数据一致，便于统一分析，将每年感染实验个体计算得出的估计育种值转换为相对育种值(RBV)，合并三年感染数据表型进行下一步的计算。

[0054] 2.1构建系谱

[0055] 根据实验室自2007年以来记录的牙鲆家系信息和亲本子代对应关系，构建牙鲆家系总系谱，涵盖用作亲本的韩国牙鲆群体(KS)、日本牙鲆群体(JS)、抗病牙鲆群体(RS)、2007年、2009年、2010年、2012年、2013年、2014年、2015年所有牙鲆亲鱼和建立的家系。

[0056] 2.2选用模型

[0057] 使用的计算模型为ASReml软件的动物模型，该模型可以同时实现固定效应育种值和随机效应育种值的最佳线性无偏预测。在估计每个个体育种值时，能够充分利用后代及亲本间一切可知的亲缘关系资料，构建亲缘关系矩阵以及逆矩阵，提高育种值估计的准确性。

[0058] $y=u+b+a+e$；$\mathrm{var}\begin{bmatrix} s \\ e \end{bmatrix}=\begin{bmatrix} A\sigma_a^2 & 0 \\ 0 & R\sigma_e^2 \end{bmatrix}$

[0059] 其中y为表型死亡时间，u为平均值，b为固定效应，a为随机效应，e为残差项。A是动

物个体间的家系遗传相关，$A\sigma_a^2$是动物个体间加性方差协方差阵。遗传混合模型的方程组为：

[0060] $$\begin{bmatrix} X'X & X'Z \\ Z'X & Z'Z+A^{-1}k \end{bmatrix}\begin{bmatrix} \hat{b} \\ \hat{a} \end{bmatrix}=\begin{bmatrix} X'y \\ Z'y \end{bmatrix}$$

[0061] 其中，$k=\frac{\sigma_s^2}{\sigma_e^2}=\ (1-h^2)\ /h^2$，遗传力的计算公式为：

[0062] $$h^2=\sigma_a^2/(\sigma_a^2+\sigma_e^2)$$

[0063] 2.3感染实验个体育种值的计算

[0064] 使用的计算软件为，R-3.22，在R环境下安装R-package asreml和AAfun，计算所用的脚本为：

[0065] library(asreml)

[0066] library(AAfun)

[0067] Mm<-asreml.read.table("data.csv",header＝T,sep＝",")

[0068] Mm.ped<-asreml.read.table("pedigree.csv",header＝T,sep＝",")

[0069] Mm.ainv<-asreml.Ainverse(Mm.ped)$ginv

[0070] ope<-asreml(fixed＝time～1+Batch,

[0071] random＝～ped(Animal),

[0072] rcov＝～units,

[0073] data＝Mm,

[0074] ginverse＝list(Animal＝Mm.ainv),

[0075] maxiter＝40)

[0076] summary(ope)$varcomp

[0077] pin(ope,h2a～V1/(V1+V2))

[0078] coef(ope)$fixed

[0079] random.effect<-coef(ope)$random

[0080] write.csv(random.effect,file＝"EBV.csv")

[0081] 通过上述脚本，分别计算2013-2015年，3组感染实验的遗传力和个体估计育种值，选择计算的性状为感染实验的个体死亡时间(小时)，存活个体按照感染结束时间记为396小时。将感染批次设为固定效应，结合2.1构建的牙鲆家系总系谱进行计算，保存个体育种值文件，根据育种均值将个体育种值转化为相对育种值，转换式：

[0082] $$RBV=(EBV+\bar{u})/\bar{u}$$

[0083] 2.4全基因组选择的参考群体的选取

[0084] 从感染实验的样品中，挑选出96个家系(2013年32个，2014年10个，2015年48个)，每个家系按照死亡率选取等比例死亡和存活个体10-15个，组成基因组选择的参考群体，将选取个体的相对育种值作为基因组选择的表型(表2)。

[0085] 表1 用于基因组重测序的牙鲆个体

[0086]

感染年份	家系数	个体数
2013	32	320
2014	10	100
2015	48	540

[0087] (二)、半滑舌鳎抗哈维氏弧菌参考群体的建立和表型的测定

[0088] 1.半滑舌鳎哈维氏弧菌感染

[0089] 2014年，建立半滑舌鳎家系共103个，当半滑舌鳎家系鱼苗生长至8-15cm时，采用腹腔接种方法对各家系鱼苗进行哈维氏弧菌感染。感染实验按照以前建立的方法进行。每个家系随机选取80-150尾鱼苗进行正式感染实验，感染用鱼苗养殖在2-3立方米的玻璃钢水槽中。感染后，每天对各家系鱼苗的状态及死亡数量进行观察记录，在病原菌感染发病后收集死亡的实验个体，记录死亡时间及全长、体重，体宽等数据，同时采集感染后存活的个体并记录其全长、体重，体宽等数据，将这些数据作为鱼体抗病力的表型参数。半滑舌鳎抗哈维氏弧菌基因组选择研究的参考群体即来源于收集了表型和鳍条的感染个体(表3)。

[0090] 表3:2014年哈维氏弧菌感染半滑舌鳎

[0091]

感染年份	家系数	个体数
2014	103	7766

[0092] 2.半滑舌鳎哈维氏弧菌感染参考群体表型性状分析

[0093] 使用分析软件ASReml-R，对参考群体表型进行分析，选用软件中的动物模型计算感染实验的遗传力和每个个体的估计育种值，并将感染实验个体计算得出的估计育种值转换为相对育种值，进行下一步的计算。

[0094] 2.1选用模型

[0095] 使用的计算模型为ASReml软件的动物模型，该模型可以同时实现固定效应育种值和随机效应育种值的最佳线性无偏预测。在估计每个个体育种值时，能够充分利用后代及亲本间一切可知的亲缘关系资料，构建亲缘关系矩阵以及逆矩阵，提高育种值估计的准确性。

[0096] $y=u+b+a+e;\ var\begin{bmatrix}s\\e\end{bmatrix}=\begin{bmatrix}A\sigma_a^2 & 0\\0 & R\sigma_e^2\end{bmatrix}$

[0097] 其中y为表型死亡时间，u为平均值，b为固定效应，a为随机效应，e为残差项。A是动物个体间的家系遗传相关，$A\sigma_a^2$是动物个体间加性方差协方差阵。遗传混合模型的方程组为：

[0098] $$\begin{bmatrix}X'X & X'Z\\Z'X & Z'Z+A^{-1}k\end{bmatrix}\begin{bmatrix}\hat{b}\\\hat{a}\end{bmatrix}=\begin{bmatrix}X'y\\Z'y\end{bmatrix}$$

[0099] 其中，$k=\frac{\sigma_s^2}{\sigma_e^2}=(1-h^2)/h^2$，遗传力的计算公式为：

[0100] $$h^2=\sigma_a^2/(\sigma_a^2+\sigma_e^2)$$

[0101] 2.2感染实验个体育种值的计算

[0102] 使用的计算软件为,R-3.22,在R中安装R-package asreml和AAfun,计算所用的脚本为:

[0103] library(asreml)

[0104] library(AAfun)

[0105] Mm<-asreml.read.table("data.csv",header＝T,sep＝",")

[0106] Mm.ped<-asreml.read.table("pedigree.csv",header＝T,sep＝",")

[0107] Mm.ainv<-asreml.Ainverse(Mm.ped)$ginv

[0108] ope<-asreml(fixed＝time～1+Batch,

[0109] random＝～ped(Animal),

[0110] rcov＝～units,

[0111] data＝Mm,

[0112] ginverse＝list(Animal＝Mm.ainv),

[0113] maxiter＝40)

[0114] summary(ope)$varcomp

[0115] pin(ope,h2a～V1/(V1+V2))

[0116] coef(ope)$fixed

[0117] random.effect<-coef(ope)$random

[0118] write.csv(random.effect,file＝"EBV.csv")

[0119] 通过上述脚本,计算2014年半滑舌鳎哈维氏弧菌感染实验群体的遗传力和个体估计育种值,选择计算的性状为感染实验的个体死亡时间,存活个体按照感染结束时间记为360小时。将感染批次设为固定效应,保存个体育种值文件,根据育种均值将个体育种值转化为相对育种值,转换式:

[0120] $$RBV = (EBV + \bar{u})/\bar{u}$$

[0121] 2.3半滑舌鳎全基因组选择参考群体的选取

[0122] 从感染实验的样品中,挑选出107个家系,每个家系按照死亡率选取等比例死亡和存活个体10个,组成基因组选择的参考群体,将选取个体的相对育种值作为基因组选择的表型(表4)。

[0123] 表4 用于基因组重测序的牙鲆个体

[0124]

感染年份	家系数	个体数
2014	107	1070

[0125] 二、参考群体的全基因组重测序及基因型采集与分析处理

[0126] 利用采集的参考群体鱼苗的鳍条提取基因组DNA,检测完整度,提取质量合格的样品用于建库测序,建库类型为DNA-350bp,测序策略为HiSeqPE150,对测序结果首先根据测序质量,过滤掉含有接头序列和低质量碱基较多的测序得到的reads,比对参考基因组,参考基因组来源于发明人完成的半滑舌鳎和牙鲆的全基因组测序结果(GenBank ID:PRJNA73987;PRJNA73673),检测参考基因组中对应的SNP位点并calling,整理生成.vcf文

件;使用Plink对.vcf文件进行读取、合并个体数据,随后按照染色体进行分割和质量控制、使用beagle进行数据填充,再将所有用到的染色体上的SNP位点数据合并,转换生成以个体为单位的基因组选择计算所使用的.csv格式基因型文件。

[0127] 下面以牙鲆和半滑舌鳎为例进行详细说明。

[0128] (一)、牙鲆全基因组重测序及基因型采集与处理分析

[0129] 1.牙鲆全基因组重测序及SNPCalling

[0130] 将选定的参考群体和候选群体鳍条送测序公司进行基因组DNA的提取和测序。基因组提取检测合格的样品共931个(表5),将这些样品建库测序,进行SNPcalling。

[0131] 表5 牙鲆基因组选择参考群体与候选群体统计

[0132]

感染年份	家系数	个体数
2013 年感染	32	297
2014 年感染	10	95
2015 年感染	48	539

[0133]

[0134] 所有样品的建库类型均为用于重测序的DNA-350bp,建库完成后,进行全基因重测序,测序策略是HiSeqPE150,数据量为2G。根据测序批次,对测序结果进行分组SNPcalling,每组的个体上限设为100个,SNPcalling使用的参考基因组来源于本实验室的牙鲆全基因组测序(GenBank ID:PRJNA73673)。SNPcalling的方法为:

[0135] 1.过滤原始数据并进行质控

[0136] 1.1过滤掉含有接头序列的reads。

[0137] 1.2当单端测序read中N的含量超过该条read长度比例的10%时,去除此对paired reads。

[0138] 1.3当单端测序read中含有的低质量(<=5)碱基数超过该条read长度比例的50%时,去除此对paired reads。

[0139] 2.参考基因组比对

[0140] 2.1使用软件BWA

[0141] 2.2比对参数:mem -t 4 -k 32 -M

[0142] 2.3过滤命令:samtools view -bS

[0143] samtools rmdup

[0144] 3.SNP位点的检测

[0145] 3.1使用软件:samtools

[0146] 3.2过滤参数:

samtools mpileup:-q 1 -C 50 -S -D -m 2 -F 0.002

(-q :skip alignments with mapQ smaller than INT [0];

-C:parameter for adjusting mapQ; 0 to disable [0];

-S:output per-sample strand bias P-value in BCF;

-D:output per-sample DP in BCF;

[0147] -m: minimum gapped reads for indel candidates [1];

-F:minimum fraction of gapped reads for candidates [0.002])

-Q 20 -d 4 -D 1000

(-Q minimum RMS mapping quality for SNPs [10] ;

-d minimum read depth [2] ;

-D maximum read depth [10000000])

[0148] 2.牙鲆基因组选择测序数据基因型的整理

[0149] 测序获得SNPcalling的结果生成的vcf文件，上传到服务器中，通过Linux系统，进行SNP位点的提取与处理，得到基因组选择计算的基因型文件。处理方法为:

[0150] 读取:通过PLINK提取vcf文件中的SNP序列，获得.ped，.map文件:

[0151] ./plink --vcf ParalichthysOlivaceus.vcf.gz --recode 12 --allow-extra-chr --out plink_1

[0152] 将.ped与.map文件合并

[0153] nohup ./plink --allow-extra-chr --file plink_1 --merge plink_1.ped plink_1.map --merge-equal-pos --recode 12 --out merge_1

[0154] 将数据按照染色体分割，并进行质量控制，质量控制阈值是基因分型率0.1;最小等位基因频率0.05;哈温平衡率0.000001。由于牙鲆有24对染色体，plink默认处理的是人的基因组，因此在分割过程中，需要将数据定义为染色体不少于24对的常见动物(如狗--dog)，代码如下:

[0155] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 1 --dog --out result_qc_chr-1 &

[0156] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 2 --dog --out result_qc_chr-2 &

[0157] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 3 --dog --out result_qc_chr-3 &

[0158] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 4 --dog --out result_qc_chr-4 &

[0159] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 5 --dog --out result_qc_chr-5 &

[0160] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 6 --dog --out result_qc_chr-6 &

[0161] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --

recode 12 --allow-extra-chr --chr 7 --dog --out result_qc_chr-7 &

[0162] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 8 --dog --out result_qc_chr-8 &

[0163] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 9 --dog --out result_qc_chr-9 &

[0164] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 10 --dog --out result_qc_chr-10 &

[0165] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 11 --dog --out result_qc_chr-11 &

[0166] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 12 --dog --out result_qc_chr-12 &

[0167] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 13 --dog --out result_qc_chr-13 &

[0168] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 14 --dog --out result_qc_chr-14 &

[0169] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 15 --dog --out result_qc_chr-15 &

[0170] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 16 --dog --out result_qc_chr-16 &

[0171] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 17 --dog --out result_qc_chr-17 &

[0172] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 18 --dog --out result_qc_chr-18 &

[0173] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 19 --dog --out result_qc_chr-19 &

[0174] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 20 --dog --out result_qc_chr-20 &

[0175] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 21 --dog --out result_qc_chr-21 &

[0176] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 22 --dog --out result_qc_chr-22 &

[0177] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 23 --dog --out result_qc_chr-23 &

[0178] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 24 --dog --out result_qc_chr-24 &

[0179] 数据填充 使用软件beagle进行数据填充，首先将上步所得的数据转换为beagle识别的文件格式：

[0180] nohup ./fcgene --ped result_qc_chr-1.ped --map result_qc_chr-1.map --oformat beagle --out plink_beagle_chr-1 &

[0181] nohup ./fcgene --ped result_qc_chr-2.ped --map result_qc_chr-2.map --oformat beagle --out plink_beagle_chr-2 &

[0182] nohup ./fcgene --ped result_qc_chr-3.ped --map result_qc_chr-3.map --oformat beagle --out plink_beagle_chr-3 &

[0183] nohup ./fcgene --ped result_qc_chr-4.ped --map result_qc_chr-4.map --oformat beagle --out plink_beagle_chr-4 &

[0184] nohup ./fcgene --ped result_qc_chr-5.ped --map result_qc_chr-5.map --oformat beagle --out plink_beagle_chr-5 &

[0185] nohup ./fcgene --ped result_qc_chr-6.ped --map result_qc_chr-6.map --oformat beagle --out plink_beagle_chr-6 &

[0186] nohup ./fcgene --ped result_qc_chr-7.ped --map result_qc_chr-7.map --oformat beagle --out plink_beagle_chr-7 &

[0187] nohup ./fcgene --ped result_qc_chr-8.ped --map result_qc_chr-8.map --oformat beagle --out plink_beagle_chr-8 &

[0188] nohup ./fcgene --ped result_qc_chr-9.ped --map result_qc_chr-9.map --oformat beagle --out plink_beagle_chr-9 &

[0189] nohup ./fcgene --ped result_qc_chr-10.ped --map result_qc_chr-10.map --oformat beagle --out plink_beagle_chr-10 &

[0190] nohup ./fcgene --ped result_qc_chr-11.ped --map result_qc_chr-11.map --oformat beagle --out plink_beagle_chr-11 &

[0191] nohup ./fcgene --ped result_qc_chr-12.ped --map result_qc_chr-12.map --oformat beagle --out plink_beagle_chr-12 &

[0192] nohup ./fcgene --ped result_qc_chr-13.ped --map result_qc_chr-13.map --oformat beagle --out plink_beagle_chr-13 &

[0193] nohup ./fcgene --ped result_qc_chr-14.ped --map result_qc_chr-14.map --oformat beagle --out plink_beagle_chr-14 &

[0194] nohup ./fcgene --ped result_qc_chr-15.ped --map result_qc_chr-15.map --oformat beagle --out plink_beagle_chr-15 &

[0195] nohup ./fcgene --ped result_qc_chr-16.ped --map result_qc_chr-16.map --oformat beagle --out plink_beagle_chr-16 &

[0196] nohup ./fcgene --ped result_qc_chr-17.ped --map result_qc_chr-17.map --oformat beagle --out plink_beagle_chr-17 &

[0197] nohup ./fcgene --ped result_qc_chr-18.ped --map result_qc_chr-18.map --oformat beagle --out plink_beagle_chr-18 &

[0198] nohup ./fcgene --ped result_qc_chr-19.ped --map result_qc_chr-19.map --oformat beagle --out plink_beagle_chr-19 &

[0199] nohup ./fcgene --ped result_qc_chr-20.ped --map result_qc_chr-20.map --oformat beagle --out plink_beagle_chr-20 &

[0200] nohup ./fcgene --ped result_qc_chr-21.ped --map result_qc_chr-21.map

--oformat beagle --out plink_beagle_chr-21 &

[0201] nohup ./fcgene --ped result_qc_chr-22.ped --map result_qc_chr-22.map --oformat beagle --out plink_beagle_chr-22 &

[0202] nohup ./fcgene --ped result_qc_chr-23.ped --map result_qc_chr-23.map --oformat beagle --out plink_beagle_chr-23 &

[0203] nohup ./fcgene --ped result_qc_chr-24.ped --map result_qc_chr-24.map --oformat beagle --out plink_beagle_chr-24 &

[0204] 使用beagle软件进行数据填充,将缺失值计为0:

[0205] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-1.bgl missing=0 out=imputed_beagle_chr-1 &

[0206] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-2.bgl missing=0 out=imputed_beagle_chr-2 &

[0207] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-3.bgl missing=0 out=imputed_beagle_chr-3 &

[0208] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-4.bgl missing=0 out=imputed_beagle_chr-4 &

[0209] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-5.bgl missing=0 out=imputed_beagle_chr-5 &

[0210] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-6.bgl missing=0 out=imputed_beagle_chr-6 &

[0211] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-7.bgl missing=0 out=imputed_beagle_chr-7 &

[0212] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-8.bgl missing=0 out=imputed_beagle_chr-8 &

[0213] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-9.bgl missing=0 out=imputed_beagle_chr-9 &

[0214] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-10.bgl missing=0 out=imputed_beagle_chr-10 &

[0215] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-11.bgl missing=0 out=imputed_beagle_chr-11 &

[0216] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-12.bgl missing=0 out=imputed_beagle_chr-12 &

[0217] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-13.bgl missing=0 out=imputed_beagle_chr-13 &

[0218] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-14.bgl missing=0 out=imputed_beagle_chr-14 &

[0219] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-15.bgl missing=0 out=imputed_beagle_chr-15 &

[0220] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-

16.bgl missing=0 out=imputed_beagle_chr-16 &

[0221] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-17.bgl missing=0 out=imputed_beagle_chr-17 &

[0222] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-18.bgl missing=0 out=imputed_beagle_chr-18 &

[0223] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-19.bgl missing=0 out=imputed_beagle_chr-19 &

[0224] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-20.bgl missing=0 out=imputed_beagle_chr-20 &

[0225] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-21.bgl missing=0 out=imputed_beagle_chr-21 &

[0226] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-22.bgl missing=0 out=imputed_beagle_chr-22 &

[0227] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-23.bgl missing=0 out=imputed_beagle_chr-23 &

[0228] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-24.bgl missing=0 out=imputed_beagle_chr-24 &

[0229] 将填充好的数据解压：

[0230] gunzip -d imputed_beagle_chr-1.plink_beagle_chr-1.bgl.phased.gz

[0231] gunzip -d imputed_beagle_chr-2.plink_beagle_chr-2.bgl.phased.gz

[0232] gunzip -d imputed_beagle_chr-3.plink_beagle_chr-3.bgl.phased.gz

[0233] gunzip -d imputed_beagle_chr-4.plink_beagle_chr-4.bgl.phased.gz

[0234] gunzip -d imputed_beagle_chr-5.plink_beagle_chr-5.bgl.phased.gz

[0235] gunzip -d imputed_beagle_chr-6.plink_beagle_chr-6.bgl.phased.gz

[0236] gunzip -d imputed_beagle_chr-7.plink_beagle_chr-7.bgl.phased.gz

[0237] gunzip -d imputed_beagle_chr-8.plink_beagle_chr-8.bgl.phased.gz

[0238] gunzip -d imputed_beagle_chr-9.plink_beagle_chr-9.bgl.phased.gz

[0239] gunzip -d imputed_beagle_chr-10.plink_beagle_chr-10.bgl.phased.gz

[0240] gunzip -d imputed_beagle_chr-11.plink_beagle_chr-11.bgl.phased.gz

[0241] gunzip -d imputed_beagle_chr-12.plink_beagle_chr-12.bgl.phased.gz

[0242] gunzip -d imputed_beagle_chr-13.plink_beagle_chr-13.bgl.phased.gz

[0243] gunzip -d imputed_beagle_chr-14.plink_beagle_chr-14.bgl.phased.gz

[0244] gunzip -d imputed_beagle_chr-15.plink_beagle_chr-15.bgl.phased.gz

[0245] gunzip -d imputed_beagle_chr-16.plink_beagle_chr-16.bgl.phased.gz

[0246] gunzip -d imputed_beagle_chr-17.plink_beagle_chr-17.bgl.phased.gz

[0247] gunzip -d imputed_beagle_chr-18.plink_beagle_chr-18.bgl.phased.gz

[0248] gunzip -d imputed_beagle_chr-19.plink_beagle_chr-19.bgl.phased.gz

[0249] gunzip -d imputed_beagle_chr-20.plink_beagle_chr-20.bgl.phased.gz

[0250] gunzip -d imputed_beagle_chr-21.plink_beagle_chr-21.bgl.phased.gz

[0251] gunzip -d imputed_beagle_chr-22.plink_beagle_chr-22.bgl.phased.gz

[0252] gunzip -d imputed_beagle_chr-23.plink_beagle_chr-23.bgl.phased.gz

[0253] gunzip -d imputed_beagle_chr-24.plink_beagle_chr-24.bgl.phased.gz

[0254] 对解压文件重命名：

[0255] mv imputed_beagle_chr-1.plink_beagle_chr-1.bgl.phased imputed_beagle_chr-1.bgl

[0256] mv imputed_beagle_chr-2.plink_beagle_chr-2.bgl.phased imputed_beagle_chr-2.bgl

[0257] mv imputed_beagle_chr-3.plink_beagle_chr-3.bgl.phased imputed_beagle_chr-3.bgl

[0258] mv imputed_beagle_chr-4.plink_beagle_chr-4.bgl.phased imputed_beagle_chr-4.bgl

[0259] mv imputed_beagle_chr-5.plink_beagle_chr-5.bgl.phased imputed_beagle_chr-5.bgl

[0260] mv imputed_beagle_chr-6.plink_beagle_chr-6.bgl.phased imputed_beagle_chr-6.bgl

[0261] mv imputed_beagle_chr-7.plink_beagle_chr-7.bgl.phased imputed_beagle_chr-7.bgl

[0262] mv imputed_beagle_chr-8.plink_beagle_chr-8.bgl.phased imputed_beagle_chr-8.bgl

[0263] mv imputed_beagle_chr-9.plink_beagle_chr-9.bgl.phased imputed_beagle_chr-9.bgl

[0264] mv imputed_beagle_chr-10.plink_beagle_chr-10.bgl.phased imputed_beagle_chr-10.bgl

[0265] mv imputed_beagle_chr-11.plink_beagle_chr-11.bgl.phased imputed_beagle_chr-11.bgl

[0266] mv imputed_beagle_chr-12.plink_beagle_chr-12.bgl.phased imputed_beagle_chr-12.bgl

[0267] mv imputed_beagle_chr-13.plink_beagle_chr-13.bgl.phased imputed_beagle_chr-13.bgl

[0268] mv imputed_beagle_chr-14.plink_beagle_chr-14.bgl.phased imputed_beagle_chr-14.bgl

[0269] mv imputed_beagle_chr-15.plink_beagle_chr-15.bgl.phased imputed_beagle_chr-15.bgl

[0270] mv imputed_beagle_chr-16.plink_beagle_chr-16.bgl.phased imputed_beagle_chr-16.bgl

[0271] mv imputed_beagle_chr-17.plink_beagle_chr-17.bgl.phased imputed_beagle_chr-17.bgl

[0272] mv imputed_beagle_chr-18.plink_beagle_chr-18.bgl.phased imputed_

beagle_chr-18.bgl

[0273] mv imputed_beagle_chr-19.plink_beagle_chr-19.bgl.phased imputed_beagle_chr-19.bgl

[0274] mv imputed_beagle_chr-20.plink_beagle_chr-20.bgl.phased imputed_beagle_chr-20.bgl

[0275] mv imputed_beagle_chr-21.plink_beagle_chr-21.bgl.phased imputed_beagle_chr-21.bgl

[0276] mv imputed_beagle_chr-22.plink_beagle_chr-22.bgl.phased imputed_beagle_chr-22.bgl

[0277] mv imputed_beagle_chr-23.plink_beagle_chr-23.bgl.phased imputed_beagle_chr-23.bgl

[0278] mv imputed_beagle_chr-24.plink_beagle_chr-24.bgl.phased imputed_beagle_chr-24.bgl

[0279] 使用R,对24对染色体上的所有数据进行合并,代码为:

[0280]
```
data1=read.table("imputed_beagle_chr-1.bgl", header=F)
data2=read.table("imputed_beagle_chr-2.bgl", header=F)
data3=read.table("imputed_beagle_chr-3.bgl", header=F)
data4=read.table("imputed_beagle_chr-4.bgl", header=F)
```

```
data5=read.table("imputed_beagle_chr-5.bgl", header=F)
data6=read.table("imputed_beagle_chr-6.bgl", header=F)
data7=read.table("imputed_beagle_chr-7.bgl", header=F)
data8=read.table("imputed_beagle_chr-8.bgl", header=F)
data9=read.table("imputed_beagle_chr-9.bgl", header=F)
data10=read.table("imputed_beagle_chr-10.bgl", header=F)
data11=read.table("imputed_beagle_chr-11.bgl", header=F)
data12=read.table("imputed_beagle_chr-12.bgl", header=F)
data13=read.table("imputed_beagle_chr-13.bgl", header=F)
data14=read.table("imputed_beagle_chr-14.bgl", header=F)
data15=read.table("imputed_beagle_chr-15.bgl", header=F)
data16=read.table("imputed_beagle_chr-16.bgl", header=F)
data17=read.table("imputed_beagle_chr-17.bgl", header=F)
data18=read.table("imputed_beagle_chr-18.bgl", header=F)
data19=read.table("imputed_beagle_chr-19.bgl", header=F)
data20=read.table("imputed_beagle_chr-20.bgl", header=F)
```

[0281]

```
data21=read.table("imputed_beagle_chr-21.bgl", header=F)
data22=read.table("imputed_beagle_chr-22.bgl", header=F)
data23=read.table("imputed_beagle_chr-23.bgl", header=F)
data24=read.table("imputed_beagle_chr-24.bgl", header=F)
data2_12=data2[-c(1,2),]
data3_12=data3[-c(1,2),]
data4_12=data4[-c(1,2),]
data5_12=data5[-c(1,2),]
data6_12=data6[-c(1,2),]
data7_12=data7[-c(1,2),]
data8_12=data8[-c(1,2),]
data9_12=data9[-c(1,2),]
data10_12=data10[-c(1,2),]
data11_12=data11[-c(1,2),]
data12_12=data12[-c(1,2),]
data13_12=data13[-c(1,2),]
data14_12=data14[-c(1,2),]
```

[0282]

```
data15_12=data15[-c(1,2),]
data16_12=data16[-c(1,2),]
data17_12=data17[-c(1,2),]
data18_12=data18[-c(1,2),]
data19_12=data19[-c(1,2),]
data20_12=data20[-c(1,2),]
data21_12=data21[-c(1,2),]
data22_12=data22[-c(1,2),]
data23_12=data23[-c(1,2),]
data24_12=data24[-c(1,2),]
data_rbind=rbind(data1,data2_12,data3_12,data4_12,data5_12,data6_12,data7_12,data8_12,data9_12,data10_12,data1
  1_12,data12_12,data13_12,data14_12,data15_12,data16_12,data17_12,data18_12,data19_12,data20_12,data21_12,
  data22_12 , data23_12,data24_12);
write.table(data_rbind,quote=F,row.names=F,col.name=F,file="imputed_rbind_all.bgl")
q()
```

[0283] 将合并文件的各位点基因型转换为012：

[0284] ./fcgene --bgl imputed_rbind_all.bgl --oformat plink --out beagle_convert_plink

[0285] ./plink --file beagle_convert_plink --geno 0.1 --maf 0.05 --hwe 0.000001 --recode A --allow-extra-chr --out final_result

[0286] 删除文件的前六列，并转换为csv格式，生成基因组选择计算的基因型文件

[0287] awk'{for(i＝7;i<＝NF;i++)printf$i""FS;print""}'final_result.raw>genotype.txt

[0288] nohup cat genotype.txt|sed's/[[:space:]]/,/g'>genotype.csv

[0289] 3.牙鲆SNP位点的统计

[0290] 整理得到的基因型文件genotype.csv，共包含SNP位点397215，分布在24个染色体上，每个染色体上SNP位点数由5223-20971不等(图1)。

[0291] (二)、半滑舌鳎参考群体全基因组重测序及基因型处理分析

[0292] 1.半滑舌鳎全基因组重测序及SNPCalling

[0293] 将选定的参考群体和候选群体鳍条送测序公司进行基因组DNA的提取和测序。基因组提取检测合格的样品共863个(表6)，将这些样品建库测序，进行SNPcalling。

[0294] 表6:牙鲆基因组选择参考群体与候选群体统计

[0295]

感染年份	家系数	个体数
2014年感染	107	863

[0296] 所有样品的建库类型均为用于重测序的DNA-350bp，建库完成后，进行全基因重测序，测序策略是HiSeqPE150，数据量为1G。根据测序批次，对测序结果进行分组SNPcalling，

每组的个体上限设为100个SNPcalling使用的参考基因组来源于本实验室的半滑舌鳎全基因组测序(GenBank ID:PRJNA73987)。SNPcalling的方法为:

[0297] 1.过滤原始数据并进行质控

[0298] 1.1过滤掉含有接头序列的reads。

[0299] 1.2当单端测序read中N的含量超过该条read长度比例的10%时,去除此对paired reads。

[0300] 1.3当单端测序read中含有的低质量(<=5)碱基数超过该条read长度比例的50%时,去除此对paired reads。

[0301] 2.参考基因组比对

[0302] 2.1使用软件BWA

[0303] 2.2比对参数:mem -t 4 -k 32 -M

[0304] 2.3过滤命令:samtools view -bS

[0305] samtools rmdup

[0306] 3.SNP位点的检测

[0307] 3.1使用软件:samtools

[0308] 3.2过滤参数:

samtools mpileup:-q 1 -C 50 -S -D -m 2 -F 0.002

(-q :skip alignments with mapQ smaller than INT [0];

-C:parameter for adjusting mapQ; 0 to disable [0];

-S:output per-sample strand bias P-value in BCF;

-D:output per-sample DP in BCF;

[0309] -m: minimum gapped reads for indel candidates [1];

-F:minimum fraction of gapped reads for candidates [0.002])

-Q 20 -d 4 -D 1000

(-Q minimum RMS mapping quality for SNPs [10] ;

-d minimum read depth [2] ;

-D maximum read depth [10000000])

[0310] 2.半滑舌鳎基因组选择测序数据基因型的处理

[0311] 从测序公司获得SNPcalling的结果生成的vcf文件,上传到服务器中,通过Linux系统,进行SNP位点的提取与处理,得到基因组选择计算的基因型文件。处理方法为:

[0312] 读取:通过PLINK提取vcf文件中的SNP序列,获得.ped,.map文件:

[0313] ./plink --vcf sole.vcf.gz --recode 12 --allow-extra-chr --out plink_1

[0314] 将.ped与.map文件合并:

[0315] nohup ./plink --allow-extra-chr --file plink_1 --merge plink_1.ped plink_1.map --merge-equal-pos --recode 12 --out merge_1

[0316] 将数据按照染色体分割,并进行质量控制,质量控制阈值是基因分型率0.1;最小等位基因频率0.05;哈温平衡率0.000001。半滑舌鳎基因组中含有20对常染色体和一对性

染色体(ZZ/ZW型)由于在基因组选择计算中,不能出现没有等位基因的位点,因此在前期处理中,需要将Z、W两条性染色体上的SNP位点剔除,只保留常染色体上的位点,代码如下:

[0317] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 1 --out result_qc_chr-1 &

[0318] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 2 --out result_qc_chr-2 &

[0319] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 3 --out result_qc_chr-3 &

[0320] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 4 --out result_qc_chr-4 &

[0321] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 5 --out result_qc_chr-5 &

[0322] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 6 --out result_qc_chr-6 &

[0323] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 7 --out result_qc_chr-7 &

[0324] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 8 --out result_qc_chr-8 &

[0325] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 9 --out result_qc_chr-9 &

[0326] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 10 --out result_qc_chr-10 &

[0327] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 11 --out result_qc_chr-11 &

[0328] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 12 --out result_qc_chr-12 &

[0329] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 13 --out result_qc_chr-13 &

[0330] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 14 --out result_qc_chr-14 &

[0331] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 15 --out result_qc_chr-15 &

[0332] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 16 --out result_qc_chr-16 &

[0333] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 17 --out result_qc_chr-17 &

[0334] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 18 --out result_qc_chr-18 &

[0335] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --

recode 12 --allow-extra-chr --chr 19 --out result_qc_chr-19 &

[0336] nohup ./plink --file merge_1 --geno 0.1 --maf 0.05 --hwe 0.000001 --recode 12 --allow-extra-chr --chr 20 --out result_qc_chr-20 &

[0337] 数据填充 使用软件beagle进行数据填充,首先将上步所得的数据转换为beagle识别的文件格式:

[0338] nohup ./fcgene --ped result_qc_chr-1.ped --map result_qc_chr-1.map --oformat beagle --out plink_beagle_chr-1 &

[0339] nohup ./fcgene --ped result_qc_chr-2.ped --map result_qc_chr-2.map --oformat beagle --out plink_beagle_chr-2 &

[0340] nohup ./fcgene --ped result_qc_chr-3.ped --map result_qc_chr-3.map --oformat beagle --out plink_beagle_chr-3 &

[0341] nohup ./fcgene --ped result_qc_chr-4.ped --map result_qc_chr-4.map --oformat beagle --out plink_beagle_chr-4 &

[0342] nohup ./fcgene --ped result_qc_chr-5.ped --map result_qc_chr-5.map --oformat beagle --out plink_beagle_chr-5 &

[0343] nohup ./fcgene --ped result_qc_chr-6.ped --map result_qc_chr-6.map --oformat beagle --out plink_beagle_chr-6 &

[0344] nohup ./fcgene --ped result_qc_chr-7.ped --map result_qc_chr-7.map --oformat beagle --out plink_beagle_chr-7 &

[0345] nohup ./fcgene --ped result_qc_chr-8.ped --map result_qc_chr-8.map --oformat beagle --out plink_beagle_chr-8 &

[0346] nohup ./fcgene --ped result_qc_chr-9.ped --map result_qc_chr-9.map --oformat beagle --out plink_beagle_chr-9 &

[0347] nohup ./fcgene --ped result_qc_chr-10.ped --map result_qc_chr-10.map --oformat beagle --out plink_beagle_chr-10 &

[0348] nohup ./fcgene --ped result_qc_chr-11.ped --map result_qc_chr-11.map --oformat beagle --out plink_beagle_chr-11 &

[0349] nohup ./fcgene --ped result_qc_chr-12.ped --map result_qc_chr-12.map --oformat beagle --out plink_beagle_chr-12 &

[0350] nohup ./fcgene --ped result_qc_chr-13.ped --map result_qc_chr-13.map --oformat beagle --out plink_beagle_chr-13 &

[0351] nohup ./fcgene --ped result_qc_chr-14.ped --map result_qc_chr-14.map --oformat beagle --out plink_beagle_chr-14 &

[0352] nohup ./fcgene --ped result_qc_chr-15.ped --map result_qc_chr-15.map --oformat beagle --out plink_beagle_chr-15 &

[0353] nohup ./fcgene --ped result_qc_chr-16.ped --map result_qc_chr-16.map --oformat beagle --out plink_beagle_chr-16 &

[0354] nohup ./fcgene --ped result_qc_chr-17.ped --map result_qc_chr-17.map --oformat beagle --out plink_beagle_chr-17 &

[0355] nohup ./fcgene --ped result_qc_chr-18.ped --map result_qc_chr-18.map --oformat beagle --out plink_beagle_chr-18 &

[0356] nohup ./fcgene --ped result_qc_chr-19.ped --map result_qc_chr-19.map --oformat beagle --out plink_beagle_chr-19 &

[0357] nohup ./fcgene --ped result_qc_chr-20.ped --map result_qc_chr-20.map --oformat beagle --out plink_beagle_chr-20 &

[0358] 使用beagle软件进行数据填充,将缺失值计为0:

[0359] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-1.bgl missing=0 out=imputed_beagle_chr-1 &

[0360] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-2.bgl missing=0 out=imputed_beagle_chr-2 &

[0361] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-3.bgl missing=0 out=imputed_beagle_chr-3 &

[0362] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-4.bgl missing=0 out=imputed_beagle_chr-4 &

[0363] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-5.bgl missing=0 out=imputed_beagle_chr-5 &

[0364] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-6.bgl missing=0 out=imputed_beagle_chr-6 &

[0365] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-7.bgl missing=0 out=imputed_beagle_chr-7 &

[0366] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-8.bgl missing=0 out=imputed_beagle_chr-8 &

[0367] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-9.bgl missing=0 out=imputed_beagle_chr-9 &

[0368] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-10.bgl missing=0 out=imputed_beagle_chr-10 &

[0369] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-11.bgl missing=0 out=imputed_beagle_chr-11 &

[0370] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-12.bgl missing=0 out=imputed_beagle_chr-12 &

[0371] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-13.bgl missing=0 out=imputed_beagle_chr-13 &

[0372] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-14.bgl missing=0 out=imputed_beagle_chr-14 &

[0373] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-15.bgl missing=0 out=imputed_beagle_chr-15 &

[0374] nohup java -Xmx1000m -jar beagle.jar unphased=plink_beagle_chr-16.bgl missing=0 out=imputed_beagle_chr-16 &

[0375] nohup java -Xmx1000m -jar beagle.jar unphased＝plink_beagle_chr-17.bgl missing＝0 out＝imputed_beagle_chr-17 &

[0376] nohup java -Xmx1000m -jar beagle.jar unphased＝plink_beagle_chr-18.bgl missing＝0 out＝imputed_beagle_chr-18 &

[0377] nohup java -Xmx1000m -jar beagle.jar unphased＝plink_beagle_chr-19.bgl missing＝0 out＝imputed_beagle_chr-19 &

[0378] nohup java -Xmx1000m -jar beagle.jar unphased＝plink_beagle_chr-20.bgl missing＝0 out＝imputed_beagle_chr-20 &

[0379] 将填充好的数据解压：

[0380] gunzip -d imputed_beagle_chr-1.plink_beagle_chr-1.bgl.phased.gz

[0381] gunzip -d imputed_beagle_chr-2.plink_beagle_chr-2.bgl.phased.gz

[0382] gunzip -d imputed_beagle_chr-3.plink_beagle_chr-3.bgl.phased.gz

[0383] gunzip -d imputed_beagle_chr-4.plink_beagle_chr-4.bgl.phased.gz

[0384] gunzip -d imputed_beagle_chr-5.plink_beagle_chr-5.bgl.phased.gz

[0385] gunzip -d imputed_beagle_chr-6.plink_beagle_chr-6.bgl.phased.gz

[0386] gunzip -d imputed_beagle_chr-7.plink_beagle_chr-7.bgl.phased.gz

[0387] gunzip -d imputed_beagle_chr-8.plink_beagle_chr-8.bgl.phased.gz

[0388] gunzip -d imputed_beagle_chr-9.plink_beagle_chr-9.bgl.phased.gz

[0389] gunzip -d imputed_beagle_chr-10.plink_beagle_chr-10.bgl.phased.gz

[0390] gunzip -d imputed_beagle_chr-11.plink_beagle_chr-11.bgl.phased.gz

[0391] gunzip -d imputed_beagle_chr-12.plink_beagle_chr-12.bgl.phased.gz

[0392] gunzip -d imputed_beagle_chr-13.plink_beagle_chr-13.bgl.phased.gz

[0393] gunzip -d imputed_beagle_chr-14.plink_beagle_chr-14.bgl.phased.gz

[0394] gunzip -d imputed_beagle_chr-15.plink_beagle_chr-15.bgl.phased.gz

[0395] gunzip -d imputed_beagle_chr-16.plink_beagle_chr-16.bgl.phased.gz

[0396] gunzip -d imputed_beagle_chr-17.plink_beagle_chr-17.bgl.phased.gz

[0397] gunzip -d imputed_beagle_chr-18.plink_beagle_chr-18.bgl.phased.gz

[0398] gunzip -d imputed_beagle_chr-19.plink_beagle_chr-19.bgl.phased.gz

[0399] gunzip -d imputed_beagle_chr-20.plink_beagle_chr-20.bgl.phased.gz

[0400] 对解压文件重命名：

[0401] mv imputed_beagle_chr-1.plink_beagle_chr-1.bgl.phased imputed_beagle_chr-1.bgl

[0402] mv imputed_beagle_chr-2.plink_beagle_chr-2.bgl.phased imputed_beagle_chr-2.bgl

[0403] mv imputed_beagle_chr-3.plink_beagle_chr-3.bgl.phased imputed_beagle_chr-3.bgl

[0404] mv imputed_beagle_chr-4.plink_beagle_chr-4.bgl.phased imputed_beagle_chr-4.bgl

[0405] mv imputed_beagle_chr-5.plink_beagle_chr-5.bgl.phased imputed_beagle_

chr-5.bgl

[0406] mv imputed_beagle_chr-6.plink_beagle_chr-6.bgl.phased imputed_beagle_chr-6.bgl

[0407] mv imputed_beagle_chr-7.plink_beagle_chr-7.bgl.phased imputed_beagle_chr-7.bgl

[0408] mv imputed_beagle_chr-8.plink_beagle_chr-8.bgl.phased imputed_beagle_chr-8.bgl

[0409] mv imputed_beagle_chr-9.plink_beagle_chr-9.bgl.phased imputed_beagle_chr-9.bgl

[0410] mv imputed_beagle_chr-10.plink_beagle_chr-10.bgl.phased imputed_beagle_chr-10.bgl

[0411] mv imputed_beagle_chr-11.plink_beagle_chr-11.bgl.phased imputed_beagle_chr-11.bgl

[0412] mv imputed_beagle_chr-12.plink_beagle_chr-12.bgl.phased imputed_beagle_chr-12.bgl

[0413] mv imputed_beagle_chr-13.plink_beagle_chr-13.bgl.phased imputed_beagle_chr-13.bgl

[0414] mv imputed_beagle_chr-14.plink_beagle_chr-14.bgl.phased imputed_beagle_chr-14.bgl

[0415] mv imputed_beagle_chr-15.plink_beagle_chr-15.bgl.phased imputed_beagle_chr-15.bgl

[0416] mv imputed_beagle_chr-16.plink_beagle_chr-16.bgl.phased imputed_beagle_chr-16.bgl

[0417] mv imputed_beagle_chr-17.plink_beagle_chr-17.bgl.phased imputed_beagle_chr-17.bgl

[0418] mv imputed_beagle_chr-18.plink_beagle_chr-18.bgl.phased imputed_beagle_chr-18.bgl

[0419] mv imputed_beagle_chr-19.plink_beagle_chr-19.bgl.phased imputed_beagle_chr-19.bgl

[0420] mv imputed_beagle_chr-20.plink_beagle_chr-20.bgl.phased imputed_beagle_chr-20.bgl

[0421] 使用R,对24对染色体上的所有数据进行合并,代码为:

[0422] data1＝read.table("imputed_beagle_chr-1.bgl",header＝F)

[0423] data2＝read.table("imputed_beagle_chr-2.bgl",header＝F)

[0424] data3＝read.table("imputed_beagle_chr-3.bgl",header＝F)

[0425] data4＝read.table("imputed_beagle_chr-4.bgl",header＝F)

[0426] data5＝read.table("imputed_beagle_chr-5.bgl",header＝F)

[0427] data6＝read.table("imputed_beagle_chr-6.bgl",header＝F)

[0428] data7＝read.table("imputed_beagle_chr-7.bgl",header＝F)

[0429] data8＝read.table("imputed_beagle_chr-8.bgl",header＝F)

[0430] data9＝read.table("imputed_beagle_chr-9.bgl",header＝F)

[0431] data10＝read.table("imputed_beagle_chr-10.bgl",header＝F)

[0432] data11＝read.table("imputed_beagle_chr-11.bgl",header＝F)

[0433] data12＝read.table("imputed_beagle_chr-12.bgl",header＝F)

[0434] data13＝read.table("imputed_beagle_chr-13.bgl",header＝F)

[0435] data14＝read.table("imputed_beagle_chr-14.bgl",header＝F)

[0436] data15＝read.table("imputed_beagle_chr-15.bgl",header＝F)

[0437] data16＝read.table("imputed_beagle_chr-16.bgl",header＝F)

[0438] data17＝read.table("imputed_beagle_chr-17.bgl",header＝F)

[0439] data18＝read.table("imputed_beagle_chr-18.bgl",header＝F)

[0440] data19＝read.table("imputed_beagle_chr-19.bgl",header＝F)

[0441] data20＝read.table("imputed_beagle_chr-20.bgl",header＝F)

[0442] data2_12＝data2[-c(1,2),]

[0443] data3_12＝data3[-c(1,2),]

[0444] data4_12＝data4[-c(1,2),]

[0445] data5_12＝data5[-c(1,2),]

[0446] data6_12＝data6[-c(1,2),]

[0447] data7_12＝data7[-c(1,2),]

[0448] data8_12＝data8[-c(1,2),]

[0449] data9_12＝data9[-c(1,2),]

[0450] data10_12＝data10[-c(1,2),]

[0451] data11_12＝data11[-c(1,2),]

[0452] data12_12＝data12[-c(1,2),]

[0453] data13_12＝data13[-c(1,2),]

[0454] data14_12＝data14[-c(1,2),]

[0455] data15_12＝data15[-c(1,2),]

[0456] data16_12＝data16[-c(1,2),]

[0457] data17_12＝data17[-c(1,2),]

[0458] data18_12＝data18[-c(1,2),]

[0459] data19_12＝data19[-c(1,2),]

[0460] data20_12＝data20[-c(1,2),]

[0461] data_rbind＝rbind(data1,data2_12,data3_12,data4_12,data5_12,data6_12,data7_12,data8_12,data9_12,data10_12,data11_12,data12_12,data13_12,data14_12,data15_12,data16_12,data17_12,data18_12,data19_12,data20_12)；

[0462] write.table(data_rbind,quote＝F,row.names＝F,col.name＝F,file＝"imputed_rbind_all.bgl")

[0463] q()

[0464] 将合并文件的各位点基因型转换为012：

[0465] ./fcgene --bgl imputed_rbind_all.bgl --oformat plink --out beagle_convert_plink

[0466] ./plink --file beagle_convert_plink --geno 0.1 --maf 0.05 --hwe 0.000001 --recode A --allow-extra-chr --out final_result

[0467] 删除文件的前六列,并转换为csv格式,生成基因组选择计算的基因型文件

[0468] awk'{for(i＝7;i<＝NF;i++)printf$i""FS;print""}'final_result.raw>genotype.txt

[0469] nohup cat genotype.txt|sed's/[[:space:]]/,/g'>genotype.csv

[0470] 3.半滑舌鳎SNP位点的统计

[0471] 整理得到的基因型文件genotype.csv,共包含SNP位点1752901,分布在20个常染色体上,每个染色体上SNP位点数由51595-200654不等(图2)。

[0472] 三、参考群体的全基因组选择计算和SNP位点效应分析

[0473] 以参考群体的相对育种值(RBV)为表型,以参考群体测序所得SNP位点数据为基因型,进行基因组选择计算,使用的计算方法为Bayes Cπ,所使用的计算工具为R-package BGLR,计算后获得每个SNP位点与抗病表型的相关性数值,整理输出.txt文件,作为各个SNP位点的遗传效应值,用于全基因组选择中的基因组育种值(GEBV)的估计。

[0474] 下面以牙鲆和半滑舌鳎为例进行详细说明。

[0475] (一)、牙鲆全基因组选择的计算和SNP位点效应分析

[0476] 1.牙鲆全基因组选择计算公式

[0477] 牙鲆基因组选择计算选用Bayes算法,这种算法着重与计算各SNP位点的效应值,然后将所有SNP位点效应值相加,得到GEBV。

[0478] SNP标记效应值的先验方差分布如下:

[0479] $$\begin{cases} \sigma_{qi}^2 = 0 & \text{先验概率 } \pi \\ \sigma_{qi}^2 > 0 & \text{先验概率}(1-\pi) \end{cases}, \quad \pi \sim U(0,1)$$

[0480] Bayes Cπ的分析模型等式为:

[0481] $$y = 1u + \sum_{i=1}^{m} X_i q_i + e$$

[0482] 模型中,y为表型值,u为群体平均值,qi是标记效应服从正态分布qi～N(0,σ_{qi}^2),m是标记的总数,X是与qi对应的关联矩阵,e是残差。

[0483] 2.牙鲆基因组选择计算方法

[0484] 使用R语言包BGLR所提供的BayesCπ算法,结合整理好的基因型数据genotype.csv与表型数据phonetype.csv,对全基因组重测序的参考群体和候选群体共988个牙鲆个体进行基因组选择计算,代码为:

[0485] sink("outofanalysis.txt")

[0486] library(bigmemory)

[0487] library(biganalytics)

[0488] library(BGLR)

[0489] pheno_y＝as.matrix(read.table("/phonetype.txt",header＝T))

[0490] y＝pheno_y[,1]

[0491] x＝as.matrix(read.big.matrix("/genotype.csv",type＝"integer",header＝T))

[0492] nIter＝22000;burnIn＝2000;thin＝100

[0493] ETA<-list(MRK＝list(X＝x,model＝"BayesC"))

[0494] fmBC<-BGLR(y＝y,ETA＝ETA,nIter＝nIter,burnIn＝burnIn,saveAt＝'BC_')

[0495] save(fmBC,file＝"/fmBC.rda")

[0496] RV＝c(fmBC$varE,fmBC$SD.varE);RV

[0497] DIC＝c(fmBC$fit);DIC

[0498] PBC＝fmBC$yhat;PBCSD＝fmBC$SD.yhat

[0499] write.table(PBC,quote＝F,row.names＝T,file＝"/PBC.txt")

[0500] ghatBC<-x%*%fmBCETAMRK$b

[0501] write.table(ghatBC,quote＝F,row.names＝T,file＝"/ghatBC.txt")

[0502] bhatBC<-fmBCETAMRK$b

[0503] SD.bhatBC<-fmBCETAMRK$SD.b

[0504] write.table(bhatBC,quote＝F,row.names＝T,file＝"/SNPBC.txt")

[0505] write.table(SD.bhatBC,quote＝F,row.names＝T,file＝"/SNPBC.SD.txt")

[0506] ryhat＝c(cor(y,fmBC$yhat))

[0507] ryhat

[0508] Rghat＝c(cor(y,ghatBC))

[0509] Rghat

[0510] ghatBC<-x%*%fmBCETAMRK$b

[0511] lm.reg＝lm(fmBC$y～ghatBC)

[0512] summary(lm.reg)

[0513] sink()

[0514] q()

[0515] 3.牙鲆SNP位点效应分析

[0516] 在988个重测序牙鲆个体进行基因组选择计算过程中,共得到397215个SNP位点的效应值,最大效应值是3.73E-4,最小效应值为1.18E-16(图3)。

[0517] (二)、半滑舌鳎全基因组选择的计算和SNP位点效应分析

[0518] 1.半滑舌鳎基因组选择计算公式

[0519] 半滑舌鳎基因组选择计算同样选用Bayes算法,SNP标记效应值的先验方差分布如下:

[0520] $$\begin{cases} \sigma_{qi}^2 = 0 & \text{先验概率}\ \pi \\ \sigma_{qi}^2 > 0 & \text{先验概率}(1-\pi) \end{cases}, \quad \pi \sim U\ (0,1)$$

[0521] Bayes Cπ的分析模型等式为:

[0522] $$y = 1u + \sum_{i=1}^{m} X_i q_i + e$$

[0523] 模型中,y为表型值,u为群体平均值,qi是标记效应服从正态分布qi～N(0,σ^2_{qi}),m是标记的总数,X是与qi对应的关联矩阵,e是残差。

[0524] 2.半滑舌鳎基因组选择计算方法

[0525] 同样使用R语言包BGLR所提供的BayesCπ算法,结合整理好的基因型数据genotype.csv与表型数据phonetype.csv,对全基因组重测序的参考群体和候选群体共886个半滑舌鳎个体进行基因组选择计算,代码为:

[0526] sink("outofanalysis.txt")

[0527] library(bigmemory)

[0528] library(biganalytics)

[0529] library(BGLR)

[0530] pheno_y=as.matrix(read.table("/phonetype.txt",header=T))

[0531] y=pheno_y[,1]

[0532] x=as.matrix(read.big.matrix("/genotype.csv",type="integer",header=T))

[0533] nIter=22000;burnIn=2000;thin=100

[0534] ETA<-list(MRK=list(X=x,model="BayesC"))

[0535] fmBC<-BGLR(y=y,ETA=ETA,nIter=nIter,burnIn=burnIn,saveAt='BC_')

[0536] save(fmBC,file="/fmBC.rda")

[0537] RV=c(fmBC$varE,fmBC$SD.varE);RV

[0538] DIC=c(fmBC$fit);DIC

[0539] PBC=fmBC$yhat;PBCSD=fmBC$SD.yhat

[0540] write.table(PBC,quote=F,row.names=T,file="/PBC.txt")

[0541] ghatBC<-x%*%fmBCETAMRK$b

[0542] write.table(ghatBC,quote=F,row.names=T,file="/ghatBC.txt")

[0543] bhatBC<-fmBCETAMRK$b

[0544] SD.bhatBC<-fmBCETAMRK$SD.b

[0545] write.table(bhatBC,quote=F,row.names=T,file="/SNPBC.txt")

[0546] write.table(SD.bhatBC,quote=F,row.names=T,file="/SNPBC.SD.txt")

[0547] ryhat=c(cor(y,fmBC$yhat))

[0548] ryhat

[0549] Rghat=c(cor(y,ghatBC))

[0550] Rghat

[0551] ghatBC<-x%*%fmBCETAMRK$b

[0552] lm.reg=lm(fmBC$y～ghatBC)

[0553] summary(lm.reg)

[0554] sink()

[0555] q()

[0556] 3.半滑舌鳎SNP位点效应分析

[0557] 使用BGLR对886个重测序半滑舌鳎个体进行基因组选择计算，共得到1752901个SNP位点的效应值，最大效应值是3.83E-5，最小效应值为3.37E-19(图4)。

[0558] 四、候选群体的建立与全基因组重测序

[0559] 候选群体是指没有进行病原菌感染、没有抗病性状表型的鱼类家系个体，从不同家系中选择一部分不知抗病表型性状的鱼类家系个体或者在基因组选择实际应用与鱼类良种培育时，选取用于繁育的亲鱼作为候选群体，采集候选群体鱼的鳍条，提取鳍条基因组DNA，进行基因组文库构建和全基因组重测序，处理分析SNP位点数据，在缺失表型值的情况下，进行基因组选择计算；在参考群体基因组选择计算完成后，通过计算出的SNP位点的效应值，结合候选群体个体各SNP位点基因型，得到候选群体基因组估计育种值。下面以牙鲆和半滑舌鳎为例进行详细说明。

[0560] (一)牙鲆抗迟缓爱德华氏菌病基因组选择候选群体的建立

[0561] 牙鲆抗迟缓爱德华氏菌基因组选择的候选群体选自2015年建立的牙鲆家系的父母本。包含其中38个家系的57个父母本个体组成候选群体，进行基因组选择计算，得到每个个体的基因组估计育种值(GEBV)，应用于牙鲆抗病良种培育中。

[0562] (二)半滑舌鳎抗哈维氏弧菌病基因组选择候选群体的建立

[0563] 半滑舌鳎抗哈维氏弧菌基因组选择候选群体选自2015年建立的半滑舌鳎家系的父母本。包含其中15个家系的23个父母本个体组成候选群体，进行基因组选择计算，得到每个个体的基因组估计育种值，应用于半滑舌鳎抗病良种培育中。

[0564] 五、参考群体和候选群体的个体基因组估计育种值(GEBV)分析

[0565] 根据获得的参考群体的每个SNP位点的遗传效应和不同个体各个SNP位点的基因型，计算得出参考群体的个体基因组估计育种值(GEBV)；在计算结束后生成并输出.txt记录参考群体个体的GEBV。

[0566] 根据参考群体个体的GEBV，初步筛选出参考群体中抗病力较强的个体，以及这些个体所在的家系。

[0567] 根据参考群体个体的SNP位点效应和候选群体的SNP基因型，进一步计算获得候选群体个体的基因组估计育种值；根据候选群体的GEBV计算结果，计算GEBV值与实际感染存活率的相关系数，进而验证基因组选择计算结果的准确性。下面以牙鲆和半滑舌鳎为例进行详细说明。

[0568] (一)、牙鲆基因组选择个体基因组估计育种值(GEBV)分析

[0569] 1.参考群体基因组估计育种值

[0570] 计算得到988个个体的基因组估计育种值(GEBV)，其中参考群体个体共931个，将参考群体个体的GEBV与ASReml计算出的相对育种值(RBV)进行相关性分析，相关系数为：0.97(图5)。

[0571] 2.候选群体基因组估计育种值

[0572] 根据参考群体计算所得SNP位点效应值和候选群体个SNP位点的基因型，同时计算出了57个候选群体样品的基因组估计育种值(表7)，根据子代的迟缓爱德华氏菌感染实验存活率进行排序，并将其GEBV进行排序，分析存活率和GEBV的相关性，相关系数为0.82。将

57个候选群体按照性别分为父本、母本两组，分别对比其子代感染存活率和GEBV排序相关系数，母本组为0.81，父本组为0.89(图6，图7)。经感染实验选育出的存活率高于80％的牙鲆家系F1551、F1503、F1501、F1550，其父母本GEBV均高于平均值，且在计算出GEBV的前20％。

[0573] 表7 2015年牙鲆家系的亲本的基因组估计育种值

[0574]

测序编号	亲本	亲本性别	2015 年建立的组合（家系）	GEBV
ET1001	F1251	F	F1501	123.5118
ET1016	F1337	M	F1503 F1505	121.7413
ET1010	F1001	F	F1550 F1551	121.7369
ET1007	F1251	F	F1522	118.9518
ET1002	F1251	F	F1503 F1504	118.6895
ET1012	F1001	F	F1554 F1555	117.9216
ET1027	F1260	F	F1502	117.259
ET1023	F1305	M	F1550	116.855
ET1045	F1323	M	F1511	115.4817
ET1024	F1317	M	F1551	115.399
ET1008	F1219	F	F1533	114.5929
ET1014	F1334	M	F1501	113.2349
ET1003	F1256	F	F1505	112.2622
ET1038	F1005	F	F1537	111.3328
ET1018	F1334	M	F1514 F1515	110.6763
ET1011	F1252	F	F1543	108.2925
ET1037	KS-79	F	F1535 F1536 F1312 F1314	108.2824
ET1026	F1422	M	F1558	107.3394
ET1021	F1421	M	F1539	105.4695
ET1055	1228	M	F1546	104.6304
ET1017	F1323	M	F1504 F1506 F1507	103.922
ET1009	F1010	F	F1539 F1540	103.7131
ET1048	F1333	M	F1529	103.5281
ET1049	F1256	M	F1535	102.5727
ET1025	F1331	M	F1554	100.7484
ET1035	F1217	F	F1529	100.2413
ET1050	F1264	M	F1536	99.44575
ET1051	F1264	M	F1537	98.13585
ET1057	1422	M	F1556	97.92809
ET1006	F1264	F	F1518 F1525	97.79947
ET1020	F1355	M	F1522	97.29576

[0575]

ET1015	F1355	M	F1502 F1525 F1532 F1533	96.23907
ET1034	F1263	F	F1526	94.84109
ET1019	F1333	M	F1517 F1518	94.36895
ET1028	F1260	F	F1507	94.12696
ET1044	F1333	M	F1509 F1510 F1513	92.29754
ET1031	F1237	F	F1515 F1546	91.38823
ET1022	F1228	M	F1543	89.33394
ET1056	F1331	M	F1552 F1553	89.09945
ET1030	F1002	F	F1510 F1511	88.20218
ET1052	F09125	M	F1540	86.29644
ET1053	F1405	M	F1544	85.48899
ET1004	F1242	F	F1514	81.9206
ET1054	F1265	M	F1541 F1547	77.14887
ET1042	F1057	F	F1552 F1553	76.98907
ET1005	F1263	F	F1517	76.52148
ET1036	F1242	F	F1532	76.3697
ET1029	F1002	F	F1508 F1509	73.91709
ET1040	F1265	F	F1541	73.84104
ET1041	F1002	F	F1544	69.55986
ET1032	F1237	F	F1516	63.17003

[0576] (二)、半滑舌鳎基因组选择个体基因组估计育种值(GEBV)分析

[0577] 1.参考群体基因组估计育种值

[0578] 计算得到886个个体的基因组估计育种值(GEBV),其中参考群体个体共963个,将参考群体个体的GEBV与ASReml计算出的相对育种值(RBV)进行相关性分析,相关系数为:0.99(图8)。

[0579] 2.候选群体基因组估计育种值

[0580] 根据参考群体计算所得SNP位点效应值和候选群体各SNP位点的基因型,同时计算出了23个候选群体样品的基因组估计育种值(表8)。

[0581] 表8 2015年半滑舌鳎家系的亲本的基因组估计育种值

[0582]

测序编号	亲本	亲本性别	建立的组合(家系)	GEBV
cs1063	海阳	F	F1504 F1508	119.9615462
cs1064	13 年家系	F	F1536 F1537 F1538	112.2379009

[0583]

cs1065	13 年家系	F	F1539 F1540	103.0953691
cs1066	13 年家系	F	F1543 F1544	111.6362749
cs1067	13 年家系	F	F1545 F1546	108.7545041
cs1068	13 年家系	F	F1547	95.30235136
cs1069	13 年家系	F	F1548 F1549	110.8912954
cs1070	13 年家系	F	F1550	106.5240249
cs1071	14L021	M	F1504	123.5321624
cs1072	14L003	M	F1508	131.2242863
cs1073	14L003	M	F1536	129.8267555
cs1074	14L019	M	F1537	107.5764415
cs1075	14L035	M	F1538	114.5897889
cs1076	14L003	M	F1539	133.1675997
cs1077	14L019	M	F1540	105.3249818
cs1078	14L003	M	F1543	133.2202046
cs1079	14L046	M	F1544	89.47059379
cs1080	14L003	M	F1545	135.910356
cs1081	14L063	M	F1546	103.5528091
cs1082	14L075	M	F1547	121.2095647
cs1083	14L040	M	F1548	94.8597309
cs1084	14L075	M	F1549	121.0954467
cs1085	14L003	M	F1550	130.7940691

[0584] 六.鱼类抗病苗种的培育

[0585] 根据全基因组选择计算得出的候选群体基因组估计育种值的大小，在表型缺失的情况下，对候选群体的抗病力进行评估和排序；选择GEBV高的候选群体个体作为亲鱼，进行抗病苗种的繁育。繁育所得的子代苗种的抗病力显著提高，在后续进行的感染实验中，存活率高于对照组，经统计，基因组选择候选群体后代的感染存活率和养殖存活率比对照组高出20%-30%。可以在养殖生产中进行推广应用。

[0586] 下面以牙鲆和半滑舌鳎为例进行详细说明。

[0587] (一)、牙鲆抗病苗种的培育

[0588] 2015年所建立的牙鲆家系，是多代选育的结果，经过自2007年开始的家系选育，选取日本牙鲆群体、韩国牙鲆群体和中国牙鲆抗蔓弧菌群体中的优良个体作为亲本，逐代以结合生长快、养殖存活率高、对鳗弧菌病抗病力强和对迟缓爱德华氏菌抗病力强等特性为目标，最终选育得到的牙鲆家系。

[0589] 在2015年牙鲆家系建立之后，进行了对比养殖，2015年11月，每个家系选取鱼苗200尾左右，做好荧光标记在同一水池混养。养殖7个月后，进行生长性状的测量和存活个体的统计，计算各家系日增重和养殖存活率。并且选取各家系牙鲆进行迟缓爱德华氏菌感染

实验，采集样品用于基因组选择计算。

[0590] 以迟缓爱德华氏菌感染牙鲆家系鱼苗为参考群体，进行SNP位点效应分析，所得结果用于候选群体GEBV的计算，候选群体中包含上述感染的各家系亲本，根据GEBV计算结果，来自F1251、F1256、F1334等家系的候选群体个体的GEBV很高或较高，其繁殖所得牙鲆子代(命名为牙鲆鲆优2号)，在迟缓爱德华氏菌感染后的存活率为74％，比对照组感染后的存活率(44％)高30％(图9)。同时鲆优2号的养殖存活率(64％)比对照组(39％)高25％(图10)。生长对比实验表明鲆优2号的生长(日增重)也比对照组快(高)(图11)。由此可见，本发明基于全基因组选择方法培育出的牙鲆鲆优2号良种具有抗病力强、养殖存活率高且生长较快的优势。

[0591] “鲆优2号”的亲本，均为牙鲆抗迟缓爱德华氏菌基因组选择计算中的候选群体个体，计算所得GEBV和家系的实际感染存活率很高或较高。由此可见，“鲆优2号”是基因组选择技术应用于鱼类抗病良种培育的一个成功事例。

[0592] (二)、半滑舌鳎抗病苗种的培育

[0593] 2014年，在完成了所有半滑舌鳎抗哈维氏弧菌基因组选择实验的参考群体的测序和GEBV的计算工作之后。根据参考群体基因组选择计算的结果，选择GEBV最高的个体所属的10个家系的雄鱼，作为2015年半滑舌鳎家系的候选亲鱼，进行单独养殖和营养强化。

[0594] 2015年9-11月，在建立半滑舌鳎家系时，选取上述雄鱼，建立半滑舌鳎家系15个，将这些家系的父本和配套的母本剪取鳍条，进行基因组DNA提取，建库测序，即为本实验中的候选群体。

[0595] 2016年8月，对2015年建立的30个家系进行哈维氏弧菌感染实验，其中有13个家系的父本为基因组选择候选群体。从感染结果可以看出，以基因组选择出的候选群体作为父本的家系，感染存活率大多较高(图12)。将参与感染的30个家系分为基因组选择父本和普通父本两个群体，其感染存活率分别为67.6％和43％，基因组选择苗种比普通苗种的存活率提高24.6％(图13)。

[0596] 感染实验的结果，证明经过基因组选择计算所筛选出候选群体作为亲本，可以有效提高半滑舌鳎养殖群体对哈维氏弧菌感染的抗病力和存活率，为半滑舌鳎抗病良种培育提供了新的、行之有效的分子育种技术手段。

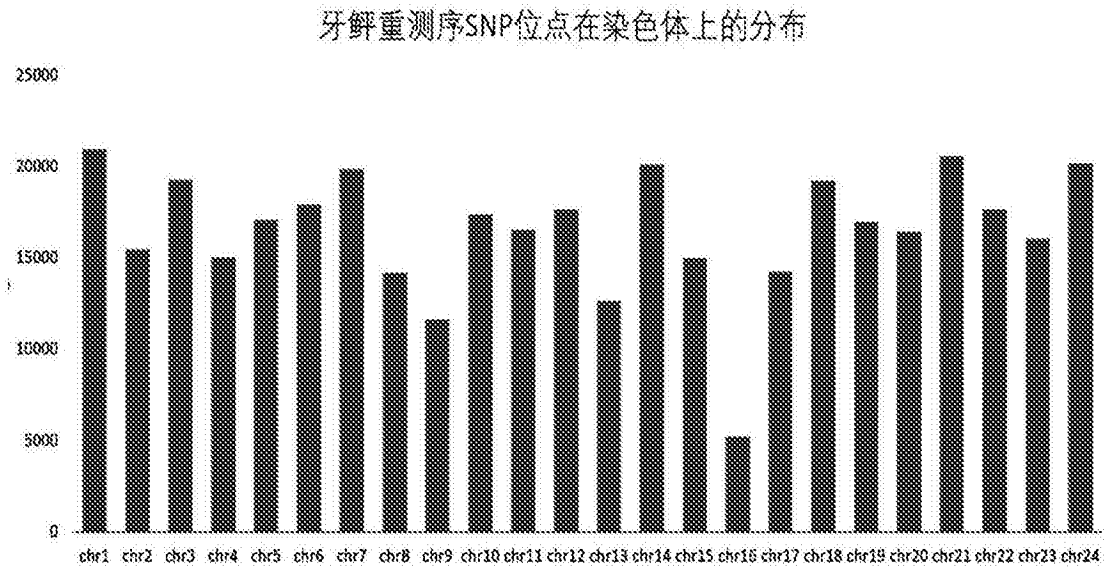

牙鲆重测序SNP位点在染色体上的分布
25000
20000
15000
10000
5000
0
chr1 chr2 chr3 chr4 chr5 chr6 chr7 chr8 chr9 chr10 chr11 chr12 chr13 chr14 chr15 chr16 chr17 chr18 chr19 chr20 chr21 chr22 chr23 chr24

图1

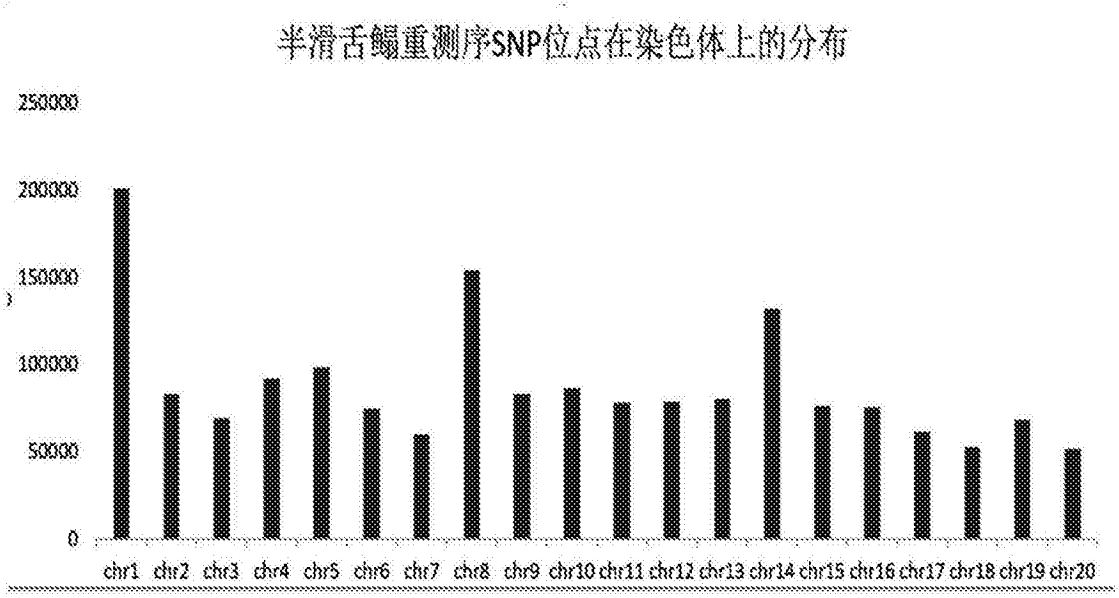

半滑舌鳎重测序SNP位点在染色体上的分布
250000
200000
150000
100000
50000
0
chr1 chr2 chr3 chr4 chr5 chr6 chr7 chr8 chr9 chr10 chr11 chr12 chr13 chr14 chr15 chr16 chr17 chr18 chr19 chr20

图2

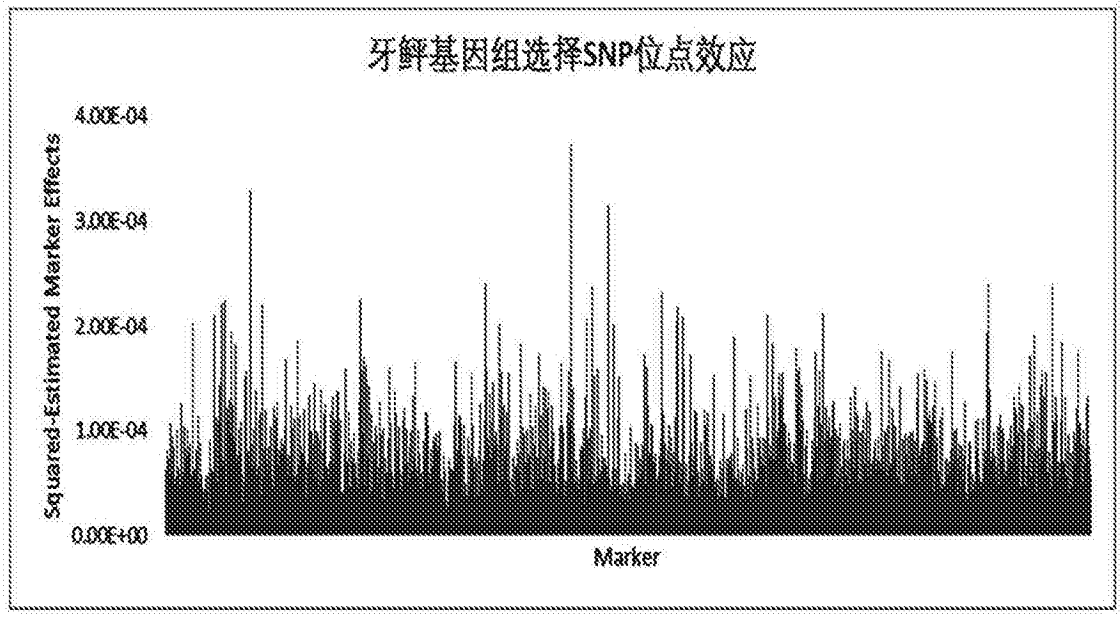

牙鲆基因组选择SNP位点效应
Squared-Estimated Marker Effects
4.00E-04
3.00E-04
2.00E-04
1.00E-04
0.00E+00
Marker

图3

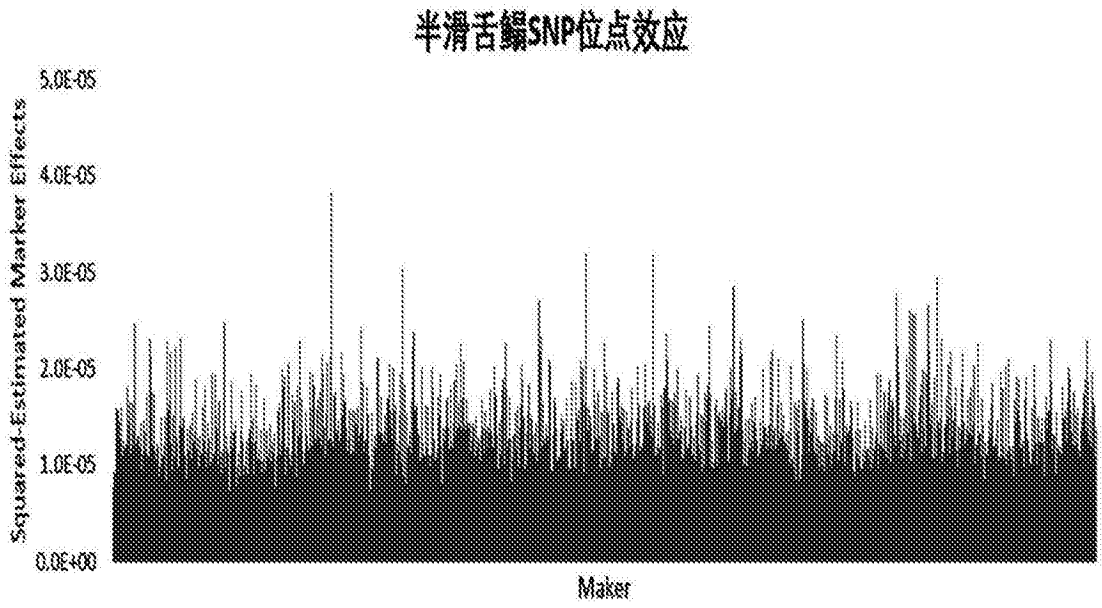

半滑舌鳎SNP位点效应
Squared-Estimated Marker Effects
5.0E-05
4.0E-05
3.0E-05
2.0E-05
1.0E-05
0.0E+00
Maker

图4

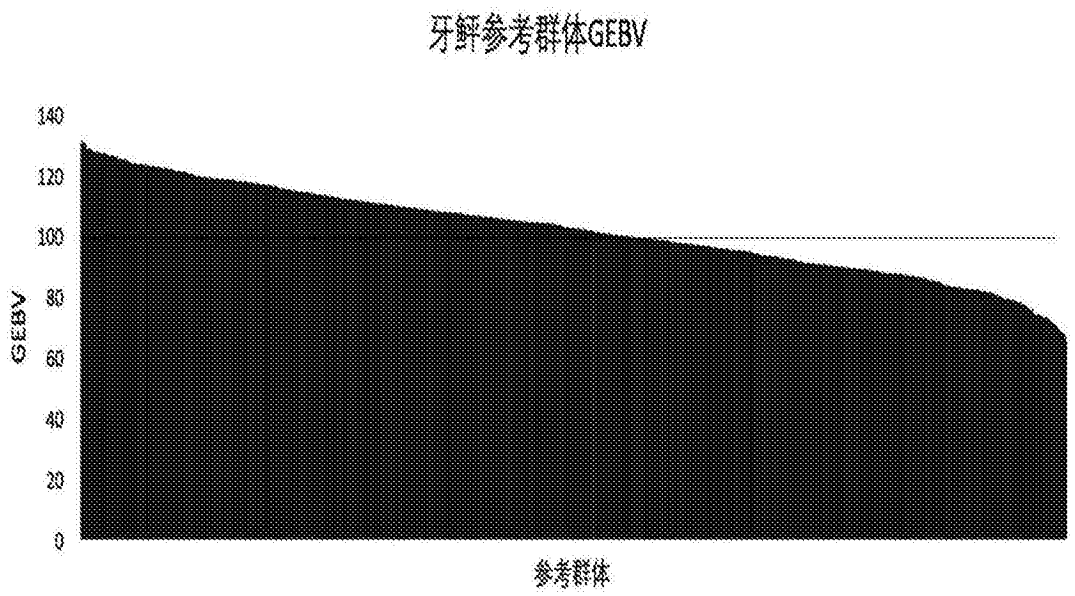

牙鲆参考群体GEBV
GEBV
140
120
100
80
60
40
20
0
参考群体

图5

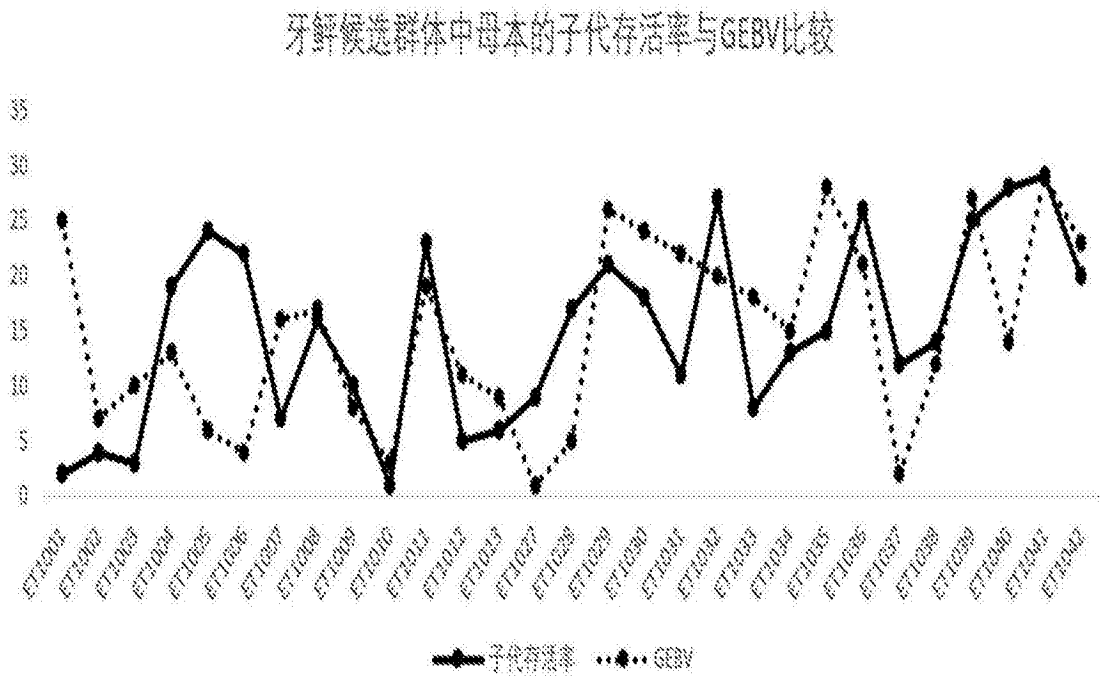

牙鲆候选群体中母本的子代存活率与GEBV比较
35
30
25
20
15
10
5
0
子代存活率
GEBV

图6

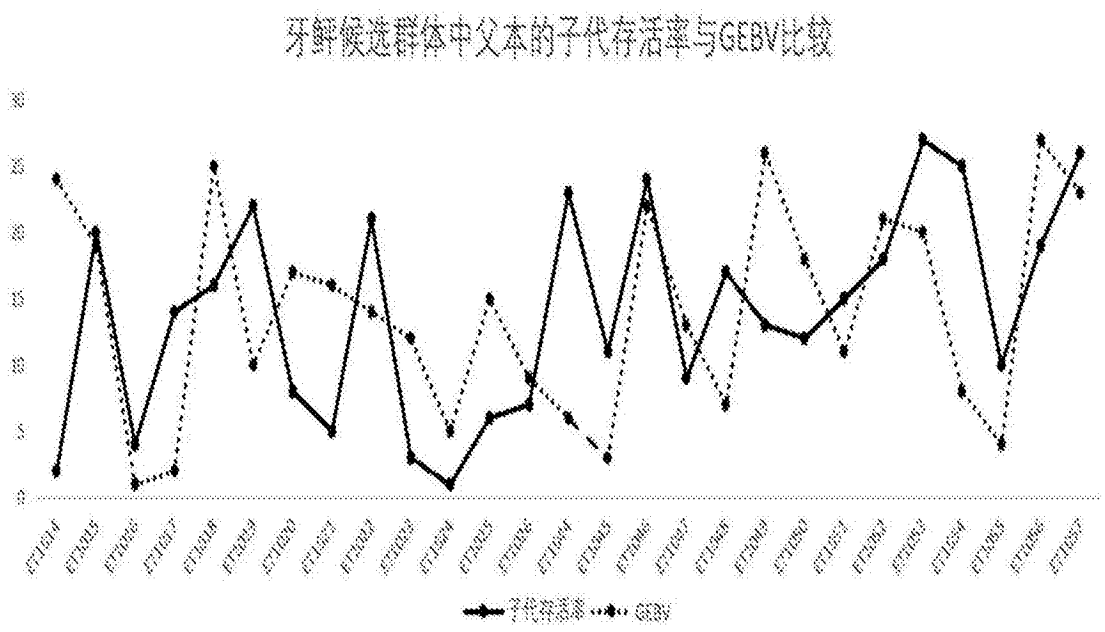

牙鲆候选群体中父本的子代存活率与GEBV比较
子代存活率
GEBV

图7

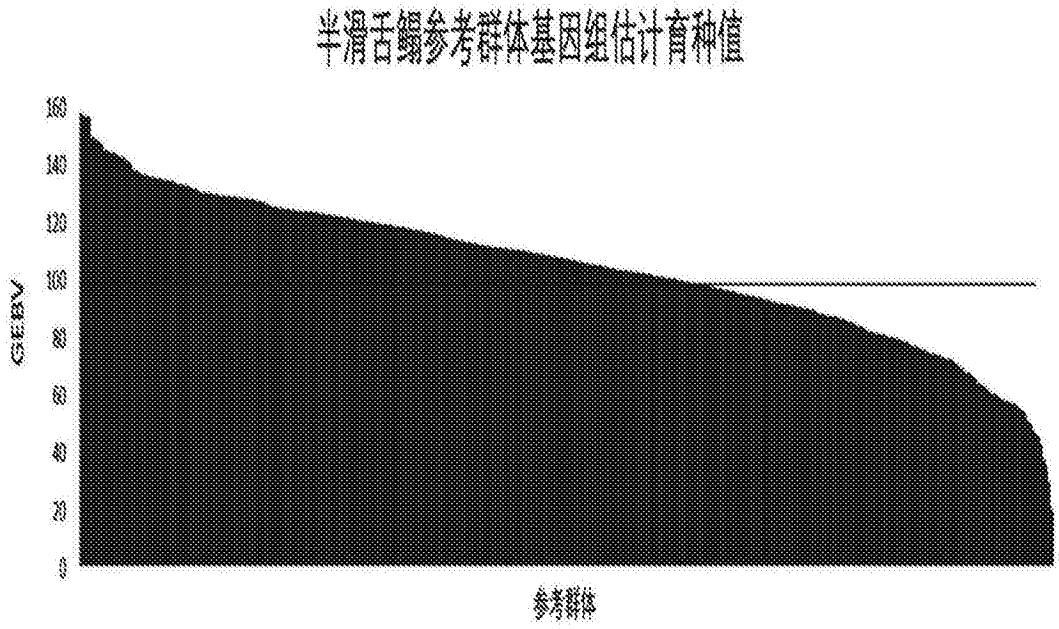

半滑舌鳎参考群体基因组估计育种值
GEBV
参考群体

图8

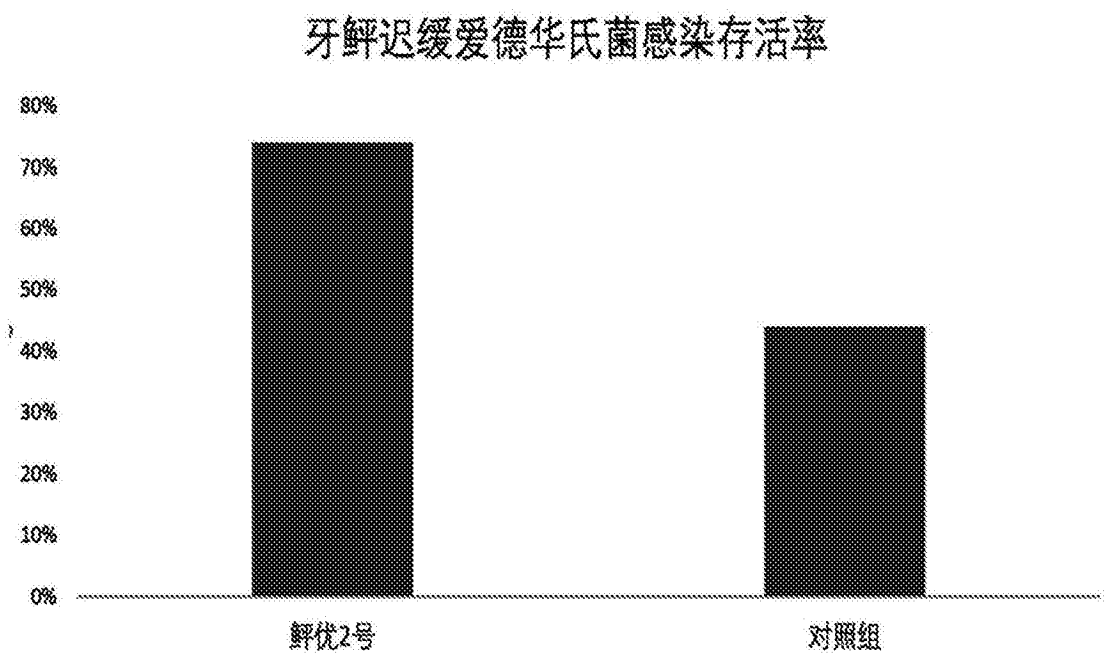

牙鲆迟缓爱德华氏菌感染存活率
80%
70%
60%
50%
40%
30%
20%
10%
0%
鲆优2号
对照组

图9

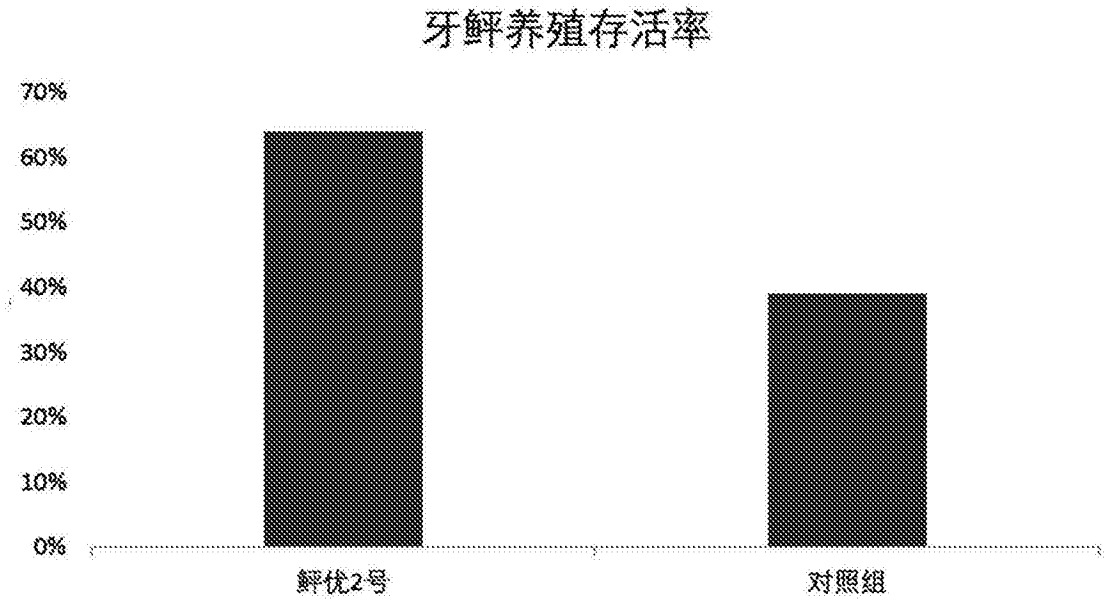

牙鲆养殖存活率
70%
60%
50%
40%
30%
20%
10%
0%
鲆优2号
对照组

图10

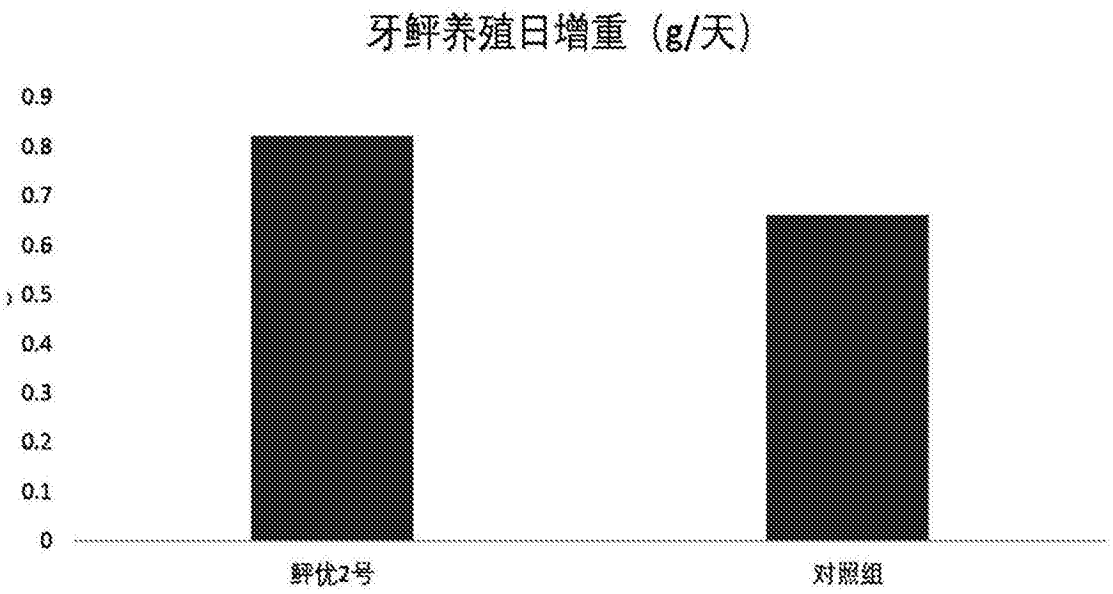

牙鲆养殖日增重（g/天）
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
鲆优2号
对照组

图11

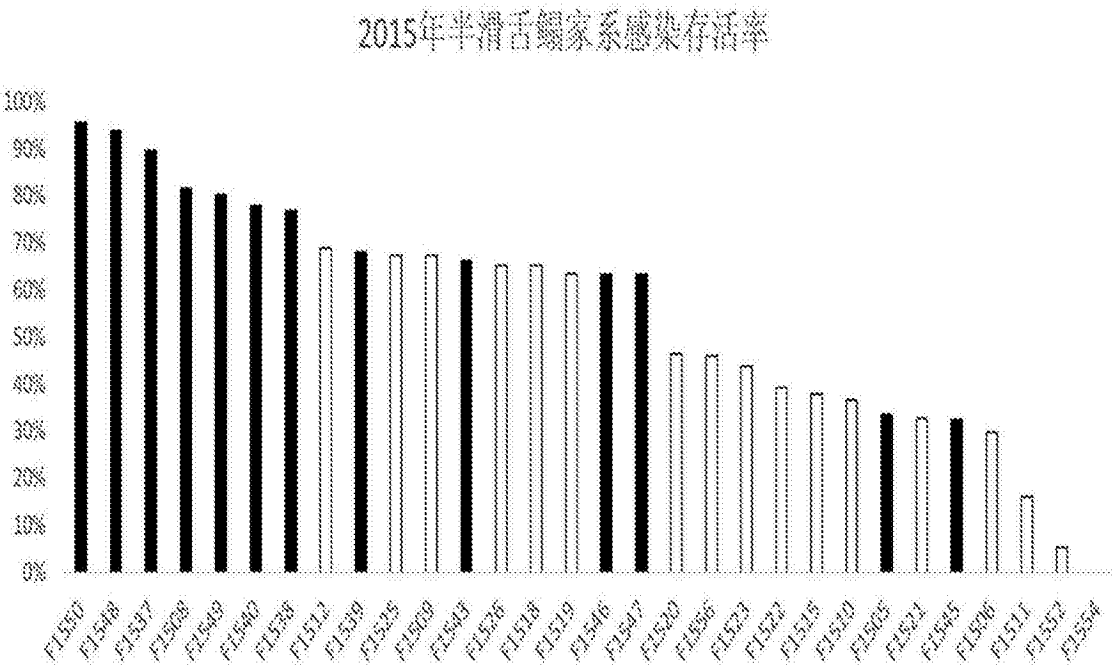

2015年半滑舌鳎家系感染存活率
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%

图12

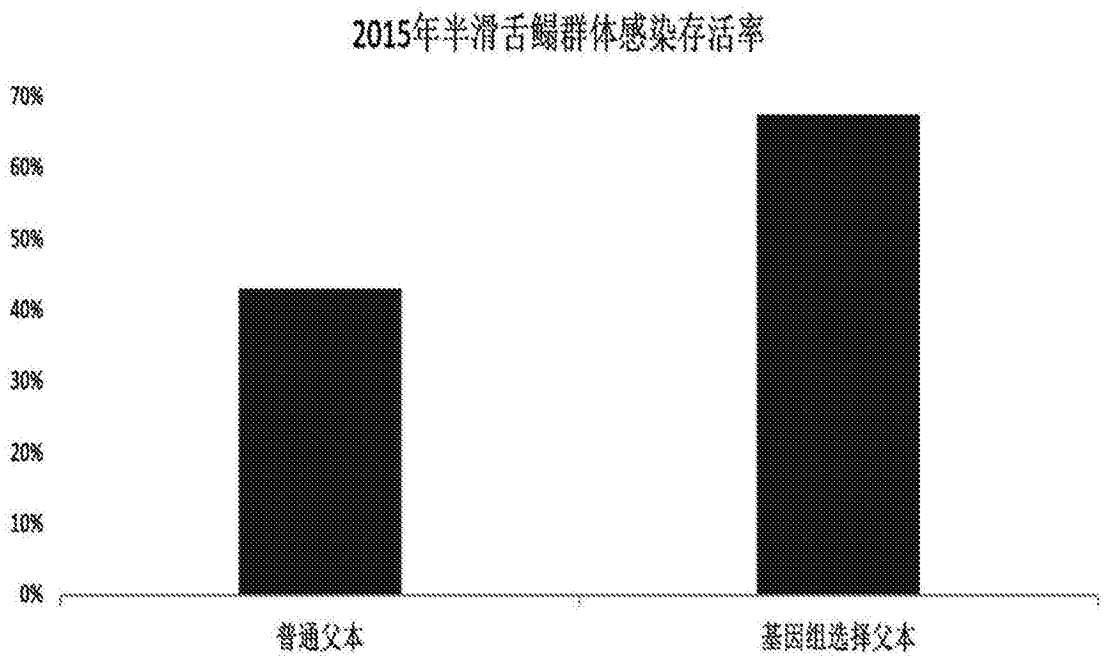


图13